

29. 3. 2004

日 本 国 特 許 庁
JAPAN PATENT OFFICE

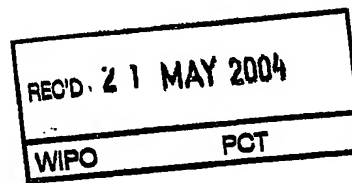
別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日 2 0 0 3 年 4 月 1 0 日
Date of Application:

出 願 番 号 特 願 2 0 0 3 - 1 0 6 3 9 3
Application Number:
[ST. 10/C]: [J P 2 0 0 3 - 1 0 6 3 9 3]

出 願 人 松 下 電 器 産 業 株 式 会 社
Applicant(s):

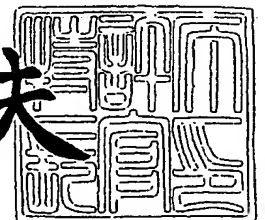


**PRIORITY
DOCUMENT**
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH RULE 17.1(a) OR (b)

2 0 0 4 年 4 月 3 0 日

特許庁長官
Commissioner,
Japan Patent Office

今 井 康 夫



【書類名】 特許願

【整理番号】 2054041182

【提出日】 平成15年 4月10日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/00 320

【発明者】

 【住所又は居所】 大阪府門真市大字門真 1 0 0 6 番地 松下電器産業株式会社内

 【氏名】 灘本 雄二

【特許出願人】

 【識別番号】 000005821

 【氏名又は名称】 松下電器産業株式会社

【代理人】

 【識別番号】 100097445

 【弁理士】

 【氏名又は名称】 岩橋 文雄

【選任した代理人】

 【識別番号】 100103355

 【弁理士】

 【氏名又は名称】 坂口 智康

【選任した代理人】

 【識別番号】 100109667

 【弁理士】

 【氏名又は名称】 内藤 浩樹

【手数料の表示】

 【予納台帳番号】 011305

 【納付金額】 21,000円

【提出物件の目録】

 【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9809938

【書類名】 明細書

【発明の名称】 ウィンドウスタック制御方法

【特許請求の範囲】

【請求項 1】 表示装置に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウスタック制御方法であって、

1つ以上のウィンドウを前記表示装置に表示する 1つ以上のアプリケーションプログラムと、

前記 1つ以上のアプリケーションプログラムが表示するウィンドウの重ね合わせを管理するウィンドウ管理プログラムとを備え、

前記アプリケーションプログラムが前記ウィンドウ管理プログラムに対してウィンドウのグループを指定し、前記ウィンドウ管理プログラムが、前記アプリケーションプログラムからウィンドウの表示要求を受け、該ウィンドウの表示を行う際に、該ウィンドウのスタック順をグループ毎に一連とする制御を行うことを特徴とするウィンドウスタック制御方法。

【請求項 2】 ウィンドウ管理プログラムは、グループ毎に纏めていない第1のウィンドウの表示要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていない全ウィンドウから、前記第1のウィンドウと同じグループに属する第1のウィンドウ群を、前記第1のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められた前記第1のウィンドウと同じグループに属する第2のウィンドウ群とスタック順を一連にし表示処理を行うことを特徴とする請求項 1 に記載のウィンドウスタック制御方法。

【請求項 3】 ウィンドウ管理プログラムは、グループ毎に代表ウィンドウを 1 枚生成し、ウィンドウのスタック順をグループ毎に一連とする際に、前記ウィンドウを前記代表ウィンドウの子ウィンドウとする事を特徴とする請求項 1 または 2 に記載のウィンドウスタック制御方法。

【請求項 4】 ウィンドウ管理プログラムは、アプリケーションプログラムからの要求を受けて、グループ内でウィンドウスタックの最上位移動、最下位移動を行うことを特徴とする請求項 1 から 3 のいずれかに記載のウィンドウスタック制御方法。

【請求項5】 ウィンドウ管理プログラムは、アプリケーションプログラムからの要求を受けて、グループ単位でウィンドウスタックの最上位移動、最下位移動を行うことを特徴とする請求項1から3のいずれかに記載のウィンドウスタック制御方法。

【請求項6】 ウィンドウ管理プログラムは、第1のウィンドウ又は第1のグループに対するスタック変更要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていない全ウィンドウから、前記第1のウィンドウと同じグループ又は前記第1のグループに属する第1のウィンドウ群を、前記第1のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められた前記第1のウィンドウと同じグループ又は前記第1のグループに属する第2のウィンドウ群とスタック順を一連にしてスタック変更を行うことを特徴とする請求項4または5に記載のウィンドウスタック制御方法。

【請求項7】 ウィンドウ管理プログラムは、アプリケーションプログラムによって指定されたウィンドウをグループに属させず、グループに属して表示された全てのウィンドウより常にスタック上位に位置させることを特徴とする請求項1に記載のウィンドウスタック制御方法。

【請求項8】 ウィンドウ管理プログラムは、最上位グループに属するウィンドウよりスタック下位のウィンドウを常に非表示状態とすることを特徴とする請求項1に記載のウィンドウスタック制御方法。

【請求項9】 ウィンドウ管理プログラムは、最上位グループに属するウィンドウの中のスタック最下位のウィンドウの直下に、特定のウィンドウを位置させることを特徴とする請求項1に記載のウィンドウスタック制御方法。

【請求項10】 ウィンドウ管理プログラムは、Xウィンドウシステムとウィンドウマネージャとを備え、最上位グループの直上に特定のウィンドウを位置させることを特徴とする請求項4または5に記載のウィンドウスタック制御方法。

【請求項11】 ウィンドウ管理プログラムは、Xウィンドウシステムとウィンドウマネージャとを備え、ウィンドウマネージャが、ウィンドウ破棄通知受信時に、ウィンドウシステム間とでスタック認識の整合性を確認し、異なっている場合は、ウィンドウシステムのスタック認識をウィンドウマネージャのスタック

認識にあわせ込む処理を行うことを特徴とする請求項 1 から 3 のいずれかに記載のウィンドウスタック制御方法。

【請求項 12】 ウィンドウ管理プログラムは、Xウィンドウシステムとウィンドウマネージャとを備え、ウィンドウマネージャが、ウィンドウシステムにスタック変更を要求する際にフラグを立てる一方で、ウィンドウ破棄通知受信時に、前記フラグが立っている時のみ、ウィンドウシステム間とでスタック認識の整合性を確認し、異なっている場合は、ウィンドウシステムのスタック認識をウィンドウマネージャのスタック認識にあわせ込む処理を行い、前記フラグを下げることを特徴とする請求項 1 から 3 のいずれかに記載のウィンドウスタック制御方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、コンピュータから表示装置に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウスタック制御方法に関する。

【0002】

【従来の技術】

近年、表示装置に複数のウィンドウを表示するコンピュータとして、ワークステーションや家庭用コンピュータが広く普及している。これらの場合、ユーザは、表示装置として比較的大きな画面に複数のウィンドウを表示して、マウスとキーボードを使ってウィンドウの重なるの上下関係であるスタック順の操作を行うのが一般的である。

【0003】

例えば、Unix (R) ワークステーションで広く普及しているX11ウィンドウシステムの場合、ウィンドウの重なるの上下関係であるスタック順は、ウィンドウの生成時にスタック最上位に位置付けられ、マップされた後は、マウスやアプリケーションの操作によって、スタック位置を上下方向に移動させる。

【0004】

具体例として、コンピュータ上で動作する 3 種類のアプリケーションが合計 5

枚のウィンドウを表示装置であるスクリーンに表示する際の表示例となる模式図を図28に示す。図28において、1はCRTモニタや液晶モニタで実現される表示装置のスクリーン、2及び3は第1のアプリケーションが表示するウィンドウ、4及び5は第2のアプリケーションが表示するウィンドウ、6は第3のアプリケーションが表示するウィンドウであり、第1のアプリケーションが表示するウィンドウ2及びウィンドウ3を第1のグループ、第2のアプリケーションが表示するウィンドウ4及びウィンドウ5を第2のグループ、第3のアプリケーションが表示するウィンドウ6を第3のグループとする。

【0005】

図28を見ればわかるように、ウィンドウ3はウィンドウ2の上に重なり、ウィンドウ5はウィンドウ4の上に重なっている。この時のウィンドウスタック順の例となる模式図を図29に示す。図29を見ればわかるように、ウィンドウ3はウィンドウ2よりスタック上位にあるので、両者が表示上で重なる場合は、ウィンドウ3がウィンドウ2の上に重なる事となり、同様に、ウィンドウ5はウィンドウ4よりスタック上位にあるので、両者が表示上で重なる場合は、ウィンドウ5がウィンドウ4の上に重なって表示される。

【0006】

そして、このような表示状態、スタック順状態で、第1のアプリケーションが何らかの都合で、例えば、ウィンドウ2をウィンドウ3の前面に表示させたくなった場合、ウィンドウ2をスタック最上位に持ってくる事で対応していた。この時のスタック順の模式図を図30に示す。図30を見ればわかるように、ウィンドウ2はウィンドウ3よりスタック上位にあるので、両者が表示上で重なる場合は、ウィンドウ2がウィンドウ3の上に重なって表示されることとなる。また、この操作により、ウィンドウ2が、第2のグループ、第3のグループのウィンドウを跨って、スタック順で下位方向から最上位へ移動した事がわかる。

【0007】

【特許文献1】

特開昭62-298882号公報

【0008】

【発明が解決しようとする課題】

しかしながら、上記従来のウィンドウスタック制御方法においては、アプリケーションが或るウィンドウに対して、ウィンドウスタック移動、ウィンドウ生成及び表示を行った場合、グループを跨ってスタック制御をするため、意図せず、該ウィンドウが他のアプリケーションの表示するウィンドウの上に重なることがあるという問題があった。

【0009】

上記課題に鑑み、本発明は、ウィンドウスタックの制御方法において、最前面でウィンドウを表示中のアプリケーションのウィンドウ表示に、他のアプリケーションが不用意に影響を及ぼす事の無いウィンドウスタック制御方法を提供することを目的とする。

【0010】**【課題を解決するための手段】**

この課題を解決するために本発明は、表示装置を含むコンピュータにおいて、前記表示装置に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウスタック制御方法であって、アプリケーションプログラムがウィンドウにグループを指定し、ウィンドウ管理プログラムがウィンドウのスタック順をグループ毎に一連とする。

【0011】

これにより、グループ最上位のウィンドウは、他のグループのウィンドウの下に重ねられて表示される事は無くなり、最前面でウィンドウを表示中のアプリケーションのウィンドウ表示に、他のアプリケーションが不用意に影響を及ぼす事を無くす事ができる。

【0012】**【発明の実施の形態】**

本発明の第1の発明は、表示装置に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウスタック制御方法であって、1つ以上のウィンドウを前記表示装置に表示する1つ以上のアプリケーションプログラムと、前記1つ以上のアプリケーションプログラムが表示するウィンドウの重ね合わせ

を管理するウィンドウ管理プログラムとを備え、前記アプリケーションプログラムが前記ウィンドウ管理プログラムに対してウィンドウのグループを指定し、前記ウィンドウ管理プログラムが、前記アプリケーションプログラムからウィンドウの表示要求を受け、該ウィンドウの表示を行う際に、該ウィンドウのスタック順をグループ毎に一連とする制御を行うことを特徴とするウィンドウスタック制御方法であり、スタック順がグループ単位で纏められるため、グループ最上位のウィンドウは、他のグループのウィンドウの下に重ねられて表示される事が無い。

【0013】

また、本発明の第2の発明は、第1の発明において、ウィンドウ管理プログラムは、グループ毎に纏めていない第1のウィンドウの表示要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていない全ウィンドウから、前記第1のウィンドウと同じグループに属する第1のウィンドウ群を、前記第1のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められた前記第1のウィンドウと同じグループに属する第2のウィンドウ群とスタック順を一連にし表示処理を行うことを特徴とするウィンドウスタック制御方法であり、表示要求によって同一グループ内でみたスタック順が変更されることがないので、アプリケーションの同一グループ内のスタック順管理の負担が軽減される。

【0014】

また、本発明の第3の発明は、第1の発明又は第2の発明において、ウィンドウ管理プログラムは、グループ毎に代表ウィンドウを1枚生成し、ウィンドウのスタック順をグループ毎に一連とする際に、前記ウィンドウを前記代表ウィンドウの子ウィンドウとする事を特徴とするウィンドウスタック制御方法であり、グループ単位のスタック移動やグループ単位のウィンドウの表示/非表示の切り替えを代表ウィンドウに対する制御で実施できる分、グループ内のウィンドウが多いほど、処理負荷を軽減することができるようになる。

【0015】

また、本発明の第4の発明は、第1の発明又は第2の発明又は第3の発明にお

いて、ウィンドウ管理プログラムは、アプリケーションプログラムからの要求を受けて、グループ内でウィンドウスタックの最上位移動、最下位移動を行うことを特徴とするウィンドウスタック制御方法であり、スタックの操作がグループ内で閉じるので、各アプリケーションプログラムは、他のアプリケーションプログラムのウィンドウ表示を気にする事無くスタック変更を行う事ができる。

【0016】

また、本発明の第5の発明は、第1の発明又は第2の発明又は第3の発明において、ウィンドウ管理プログラムは、アプリケーションプログラムからの要求を受けて、グループ単位でウィンドウスタックの最上位移動、最下位移動を行うことを特徴とするウィンドウスタック制御方法であり、ウィンドウの操作をグループ指定で実施できるので、例えば、グループをアプリケーションプログラム単位で振り分けた場合だと、そのアプリケーションに属するウィンドウのメンバーを知らなくても、スタック最前面にそのアプリケーションに属する全通常ウィンドウを移動させる事が可能となる。

【0017】

また、本発明の第6の発明は、第4の発明又は第5の発明において、ウィンドウ管理プログラムは、第1のウィンドウ又は第1のグループに対するスタック変更要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていない全ウィンドウから、前記第1のウィンドウと同じグループ又は前記第1のグループに属する第1のウィンドウ群を、前記第1のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められた前記第1のウィンドウと同じグループ又は前記第1のグループに属する第2のウィンドウ群とスタック順を一連にしてスタック変更を行うことを特徴とするウィンドウスタック制御方法であって、スタック変更対象となるグループに属する通常ウィンドウ全てを、その中のスタック順は保持したままで纏めた上でスタック変更を実施するので、アプリケーションプログラムは、自グループのウィンドウのスタック順を把握し易くなる。

【0018】

また、本発明の第7の発明は、第1の発明において、ウィンドウ管理プログラ

ムは、アプリケーションプログラムによって指定されたウィンドウをグループに属させず、グループに属して表示された全てのウィンドウより常にスタック上位に位置させることを特徴とするウィンドウスタック制御方法であって、グループ単位のスタック順に関係無くウィンドウを表示できるので、最優先で表示するウィンドウのサービスを提供することができる。

【0019】

また、本発明の第8の発明は、第1の発明において、ウィンドウ管理プログラムは、最上位グループに属するウィンドウよりスタック下位のウィンドウを常に非表示状態とすることを特徴とするウィンドウスタック制御方法であって、最上位グループよりスタック下位のグループが何を表示しようとしても表示装置には表示されないので、最上位グループのウィンドウで覆われない部分に中途半端に映り込む心配が無い。

【0020】

また、本発明の第9の発明は、第1の発明において、ウィンドウ管理プログラムは、最上位グループに属するウィンドウの中のスタック最下位のウィンドウの直下に、特定のウィンドウを位置させることを特徴とするウィンドウスタック制御方法であって、例えば、前記特定のウィンドウを表示装置のスクリーン一杯に表示しておけば、前記特定のウィンドウのスタック下位のグループが何を表示しようとしても、表示装置に表示されないので、最上位グループのウィンドウで覆われない部分に中途半端に映り込む心配が無い。

【0021】

また、本発明の第10の発明は、第4の発明又は第5の発明において、ウィンドウ管理プログラムは、Xウィンドウシステムとウィンドウマネージャとを備え、最上位グループの直上に特定のウィンドウを位置させることを特徴とするウィンドウスタック制御方法であって、最上位グループを別のグループに入れ替える際、Xウィンドウシステムの場合、XRestackWindows (R) 関数を使用する事となるが、ウィンドウ群のスタック移動先を示すために使用できる。もし、最上位グループのウィンドウよりスタック上位にウィンドウが存在しなかった場合、存在する全てのウィンドウの並び順を決め、前記XRestackWindows (R) の引数に代

入れねばならないが、前記特定のウィンドウがあれば、前記特定のウィンドウと前記ウィンドウ群を引数に代入するだけで済む。

【0022】

また、本発明の第11の発明は、第1の発明又は第2の発明又は第3の発明において、ウィンドウ管理プログラムは、Xウィンドウシステムとウィンドウマネージャとを備え、ウィンドウマネージャが、ウィンドウ破棄通知受信時に、ウィンドウシステム間とでスタック認識の整合性を確認し、異なっている場合は、ウィンドウシステムのスタック認識をウィンドウマネージャのスタック認識にあわせ込む処理を行うことを特徴とするウィンドウスタック制御方法であって、ウィンドウ破棄通知受信時に整合性確保を図るので、ウィンドウシステムがスタック変更処理に失敗しても、処理が破綻する事がなくなる。

【0023】

また、本発明の第12の発明は、第1の発明又は第2の発明又は第3の発明において、ウィンドウ管理プログラムは、Xウィンドウシステムとウィンドウマネージャとを備え、ウィンドウマネージャが、ウィンドウシステムにスタック変更を要求する際にフラグを立てる一方で、ウィンドウ破棄通知受信時に、前記フラグが立っている時のみ、ウィンドウシステム間とでスタック認識の整合性を確認し、異なっている場合は、ウィンドウシステムのスタック認識をウィンドウマネージャのスタック認識にあわせ込む処理を行い、前記フラグを下げることを特徴とするウィンドウスタック制御方法であって、ウィンドウマネージャがスタック変更を実施した時のみ、スタック認識の整合性のための処理を行うので、処理の実行頻度を軽減できる。

【0024】

以下、本発明の実施の形態について、図面を用いて説明する。

【0025】

(実施の形態1)

本実施の形態では、ウィンドウ管理プログラムが、Xウィンドウシステムとウィンドウマネージャから成り、ウィンドウマネージャがルートウィンドウの子ウィンドウであるトップレベルウィンドウのスタック順を制御する場合を例にとつ

て説明する。

【0026】

図1に本発明の実施の形態1のウィンドウスタック制御方法を実現するシステムの構成例を示す。

【0027】

図1において、7はコンピュータ、8はCPU (Central Processing Unit)、9は各種プログラムや処理データ等を格納するメモリ、10は各構成要素に所望の双方向通信を行わせるバス、11はCRTモニタや液晶モニタといった表示装置、1は表示装置11のスクリーン、101及び102、103はCPU8で実行され1枚以上のウィンドウをウィンドウ管理プログラム104を介してスクリーン1に表示しようとするアプリケーションプログラム、104はCPU8で実行されアプリケーションプログラムが表示しようとするウィンドウを管理してバス10を介して表示を行うウィンドウ管理プログラムであり、ウィンドウシステム12とウィンドウマネージャ105から成る、ウィンドウシステム12は従来のXウィンドウシステム、105はウィンドウマネージャである。

【0028】

以下、図面を用いて本実施の形態の動作を説明する。

【0029】

まず、アプリケーションプログラム101、102、103は、ウィンドウ管理プログラム104を通じて、スタック制御特性の異なる2種類のトップレベルウィンドウを表示することができるものとし、その2種類を通常ウィンドウ、優先ウィンドウと呼ぶ事とする。両者の違いは、表示された優先ウィンドウは表示された全ての通常ウィンドウより常にスタック上位に表示されることと、通常ウィンドウはグループ単位でスタック順を一連とするが優先ウィンドウでは行わないことである。

【0030】

アプリケーションプログラム101、102、103は、ウィンドウを表示する場合、ウィンドウシステム12に対して、ウィンドウ生成要求、ウィンドウ種類及びグループ設定要求、マップ要求を発行する。この内、ウィンドウ種類及び

グループ設定要求については、Xウィンドウシステムの場合、プロパティ機構を用いる事で、ウィンドウ毎に値を設定することができる。

【0031】

各ウィンドウは、ウィンドウ生成及びマップという過程を経て初めてスクリーン1に表示されるようになる。

【0032】

ウィンドウシステム12は、ウィンドウ生成要求を受け取ると、ウィンドウの重なりであるスタック順の最上位にそのウィンドウを位置させ、ウィンドウマネージャ105に対してウィンドウ生成通知イベント(CreateNotify)を発行する。

【0033】

ウィンドウマネージャ105は、内部データベースとして、ウィンドウ毎にウィンドウ情報を保持する。

【0034】

(表1) にウィンドウ情報の一例を示す。

【0035】

【表1】

情報名	内容
ウィンドウ識別子	識別子
スタック位置確定フラグ	暫定/確定済み
スタック位置情報	位置情報
プロパティ獲得フラグ	未獲得/獲得済み
ウィンドウ種類	優先ウィンドウ/通常ウィンドウ
グループ値	グループの値

【0036】

表1に示されるように、ウィンドウ情報は、ウィンドウ識別子、スタック位置確定フラグ、スタック位置情報、プロパティ獲得フラグ、ウィンドウ種類、グル

ープ値から成る。ウィンドウ識別子は、そのウィンドウ情報がどのウィンドウの情報かを示し、スタック位置確定フラグは、そのウィンドウが、ウィンドウマネージャにとって、スタック位置が暫定か確定済みなのかを示し、スタック位置情報は、そのウィンドウのスタック位置を示し、プロパティ獲得フラグは、そのウィンドウのウィンドウ種類とグループ値をウィンドウマネージャが獲得済みかどうかを示し、ウィンドウ種類は、前記プロパティ獲得フラグが獲得済みとなっている場合は、そのウィンドウが優先ウィンドウか通常ウィンドウのどちらであるかを示し、グループ値は、前記プロパティ獲得フラグが獲得済みとなっている場合は、そのウィンドウのグループ値を示す。

【0037】

図1のウィンドウマネージャ105は、ウィンドウ生成通知イベント(CreateNotify)を受け取ることで、新規にウィンドウが生成された事を知り、そのウィンドウ生成通知イベント(CreateNotify)からウィンドウ識別子を獲得し、内部データベースに新規にウィンドウ情報格納領域を設け、そのウィンドウ情報のウィンドウ識別子に格納し、また、同ウィンドウのスタック位置確定フラグを暫定に、スタック位置情報をスタック最上位に、プロパティ獲得フラグを未獲得に、各々初期化する。

【0038】

次に、ウィンドウシステム12は、ウィンドウ種類及びグループ設定要求を受け取り、該ウィンドウのプロパティとして記憶する。

【0039】

次に、ウィンドウシステム12は、アプリケーションプログラム101や102、103からマップ要求を受け取り、内部処理は特にせずに、ウィンドウマネージャ105にマップ要求イベント(MapRequest)を発行する。

【0040】

マップ要求イベント(MapRequest)を受け取った際のウィンドウマネージャ105の動作例となるフロー図を図2に示す。

【0041】

図2に示されるように、S201では、該時点で受け取ったマップ要求イベント(M

apRequest)からマップ対象ウィンドウの識別子を獲得して、S202に進む。

【0042】

S202では、マップ対象ウィンドウの識別子で内部データベースを検索し、対応するウィンドウ情報が存在する場合はS203へ進み、存在しない場合はS214へ進む。

【0043】

S203では、内部データベースで対象ウィンドウのスタック位置確定フラグが確定済みかどうかをチェックし、確定済みの場合はS213に進み、暫定の場合はS204に進む。

【0044】

S204では、対象ウィンドウのプロパティ獲得フラグをチェックし、獲得済みの場合はS206に進み、未獲得の場合はS205に進む。

【0045】

S205では、対象ウィンドウのウィンドウ種類及びグループ値をウィンドウシステムに問い合わせ獲得し、内部データベースに格納し、S206に進む。

【0046】

S206では、対象ウィンドウのウィンドウ種類をチェックし、優先ウィンドウの場合は、S213に進み、通常ウィンドウの場合は、S207に進む。

【0047】

S207では、内部データベースに、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが確定済みの通常ウィンドウがあるかどうかをチェックし、有る場合はS208に進み、無い場合はS209に進む。

【0048】

S208では、対象ウィンドウが、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが確定済みの通常ウィンドウの内のスタック最上位のウィンドウの直上となるようウィンドウシステム12にスタック変更要求を発行する。

【0049】

S208のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図3に示す。

【0050】

図3に示されるように、通常ウィンドウである対象ウィンドウが、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが確定済みのウィンドウの内のスタック最上位のウィンドウの直上に移されているのが解る。S209では、内部データベースから、通常ウィンドウ全体を対象にスタック位置確定フラグが、確定済みのウィンドウを検索し、存在する場合はS210に進み、存在しない場合はS211に進む。

【0051】

S210では、内部データベースにあるスタック位置確定フラグが確定済みの通常ウィンドウのうち、スタック位置が最上位のウィンドウの直上に対象ウィンドウが位置するよう、ウィンドウシステム12に対してスタック変更要求を発行し、S212に進む。

【0052】

S210のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図4に示す。

図4に示されるように、通常ウィンドウである対象ウィンドウが、スタック位置確定フラグが確定済みの通常ウィンドウ全体の内のスタック最上位のウィンドウの直上に移されているのが解る。

【0053】

S211では、スタック位置の最下位に対象ウィンドウが位置するよう、ウィンドウシステム12にスタック変更要求を発行し、S212に進む。

【0054】

S211のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図5に示す。

図5に示されるように、通常ウィンドウである対象ウィンドウが、スタック最下位に移されているのが解る。

【0055】

S212では、ウィンドウシステム12に対してマップ対象ウィンドウのマップ要求を発行し、内部データベースの対象ウィンドウのウィンドウ情報の内、スタック

ク位置確定フラグが暫定を示している場合は確定済みに更新し、また、内部データベース全体に対して必要に応じてスタック位置情報を更新した上で、S213に進む。

【0056】

S213では、処理を終了する。

【0057】

ウィンドウシステム12は、図2のS208やS210、S211に起因するスタック変更要求をウィンドウマネージャ105から受け取り、要求に従ってウィンドウのスタック順を変更する。また、図2のS212に起因するマップ要求をウィンドウマネージャ105から受け取り、要求に従ってウィンドウのマップする。マップされたウィンドウは、ウィンドウの重なり具合にもよるが、他のウィンドウの下に隠れていなければ、スクリーン1に表示されるようになる。

【0058】

図2の処理によって、アプリケーション101、102、103が生成及びウィンドウ種類及びグループ設定した通常ウィンドウは、表示までに必ず、スタック位置情報が確定済みとなり、また、スタック位置がグループ毎に纏められ、また、優先ウィンドウのスタック上位に表示される事もない。また、優先ウィンドウは、スタック位置が変更されることなく表示される。

【0059】

図1において、アプリケーションプログラム101、102、103は、生成及びウィンドウ種類及びグループ設定した通常ウィンドウに対して、スタック変更要求として、グループ内スタック最上位移動をウィンドウシステム12に対して発行する事ができる。Xウィンドウシステムの場合、例えば、グループ内スタック最上位移動要求にXRaiseWindow()関数を割り当てれば良い。また、優先ウィンドウに対して、スタック変更要求として、優先ウィンドウスタック最上位移動要求をウィンドウシステム12に対して発行する事ができ、例えば、グループ内スタック最上位移動要求と同じXRaiseWindow()関数を割り当てる。

【0060】

アプリケーションプログラムがXRaiseWindow()を実施した場合、ウィンドウシ

システム 12 はスタック変更をせず、ウィンドウマネージャ 105 に対してスタック最上位移動要求イベント (ConfigureRequest) を発行する。

【0061】

スタック最上位移動要求イベントを受け取った際のウィンドウマネージャ 105 の動作例となるフロー図を図 6 に示す。

【0062】

図 6 に示されるように、S601 では、該時点で受け取ったスタック最上位移動要求イベント (ConfigureRequest) から対象ウィンドウの識別子を獲得して、S602 に進む。

【0063】

S602 では、対象ウィンドウの識別子で内部データベースを検索し、対応するデータが存在する場合は S603 へ進み、存在しない場合は S612 へ進む。

【0064】

S603 では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム 12 に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値とプロパティ獲得済みである事を内部データベースに記憶し、S604 に進む。

【0065】

S604 では、対象ウィンドウのウィンドウ種類をチェックし、優先ウィンドウであれば S605 に進み、通常ウィンドウであれば S606 に進む。

【0066】

S605 では、内部データベースからウィンドウ種類が優先ウィンドウであるものの内、スタック最上位のウィンドウを検索し、そのウィンドウが対象ウィンドウ自身であれば、何もせず S612 に進み、そのウィンドウが対象ウィンドウ自身でなければ、そのウィンドウの直上となるよう、ウィンドウシステム 12 にスタック変更要求を発行し、次に、内部データベース全体に対してスタック位置情報を更新した上で、S612 に進む。

【0067】

S605のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図7に示す。

【0068】

図7に示されるように、対象ウィンドウのみが優先ウィンドウの中のスタック最上位のウィンドウの直上に移されているのが解る。

S606では、対象ウィンドウと同じグループに属する通常ウィンドウを内部データベースから検索してリストアップし、S607に進む。

【0069】

S607では、前記リストアップした中にスタック位置確定フラグが確定済みであるウィンドウが存在するならばS608に進み、存在しないならS609に進む。

【0070】

S608では、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば、それら全てを、それらの中でみたスタック順はそのまま、リストアップした中でスタック位置確定フラグが確定済みであるウィンドウの内の最上位のウィンドウの直上に位置させ、更に対象ウィンドウをリストアップしたウィンドウの中で最上位となるよう、ウィンドウシステム12にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S610に進む。S608のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図8に示す。

【0071】

図8に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定のウィンドウが、同じグループ値を持つスタック位置確定フラグが確定済みのウィンドウの直上に移され、同一グループ値を持つウィンドウ全体の中で対象ウィンドウのスタック最上位移動がなされているのが解る。

【0072】

図6において、S609では、内部データベースにスタック位置確定フラグが確定済みのウィンドウが存在するならばS610に進み、存在しないならばS611に進む。

【0073】

S610では、S606でリストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのまま、内部データベースの中でスタック位置確定フラグが確定済みであるスタック最上位の通常ウィンドウの直上に位置させ、更に対象ウィンドウを前記リストアップしたウィンドウの中で最上位となるよう、ウィンドウシステム12にスタック変更要求を発行し、次に、リストアップしたウィンドウのスタック位置確定フラグを確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S612に進む。

【0074】

S610のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図9に示す。

【0075】

図9に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック位置確定フラグが確定済みの通常ウィンドウ全体のスタック最上位ウィンドウの直上に移され、同一グループ値を持つウィンドウ全体の中で対象ウィンドウのスタック最上位移動がなされているのが解る。

【0076】

図6において、S611では、S606でリストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのまま、全体スタックの最下位とし、更に対象ウィンドウをリストアップしたウィンドウの中で最上位となるよう、ウィンドウシステム12にスタック変更要求を発行し、次に、リストアップしたウィンドウのスタック位置確定フラグを確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S612に進む。

【0077】

S611のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図10に示す。

【0078】

図10に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック最下位に移され、同一グループ値を持つ通常ウィンドウ全体の中で対象ウィンドウのスタック最上位移動がなされているのが解る。

【0079】

図6において、S612では、処理を終了する。

【0080】

図1において、アプリケーションプログラム101、102、103は、生成及びウィンドウ種類及びグループ設定したウィンドウに対して、グループ内スタック最下位移動をウィンドウシステム12に対して発行する事ができる。Xウィンドウシステムの場合、例えば、グループ内スタック最下位移動にXLowerWindow()関数を割り当てれば良い。また、優先ウィンドウに対して、スタック変更要求として、優先ウィンドウスタック最下位移動要求をウィンドウシステム12に対して発行する事ができ、例えば、グループ内スタック最下位移動要求と同じXLowerWindow()関数を割り当てる。

【0081】

アプリケーションプログラムがXLowerWindow()を実施した場合、ウィンドウシステム12はスタック変更をせず、ウィンドウマネージャ105に対してスタック最下位移動要求イベント(ConfigureRequest)を発行する。スタック最下位移動要求イベントを受け取った際のウィンドウマネージャ105の動作例となるフロー図を図11に示す。

【0082】

図11に示されるように、S1101では、該時点で受け取ったスタック最下位移動要求イベント(ConfigureRequest)から対象ウィンドウの識別子を獲得して、S1102に進む。

【0083】

S1102では、対象ウィンドウの識別子で内部データベースを検索し、対応するデータが存在する場合はS1103へ進み、存在しない場合はS1112へ進む。

【0084】

S1103では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム12に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値とプロパティ獲得済みである事を内部データベースに記憶し、S1104に進む。

【0085】

S1104では、対象ウィンドウのウィンドウ種類をチェックし、優先ウィンドウの場合はS1105へ進み、通常ウィンドウの場合はS1106へ進む。

【0086】

S1105では、内部データベースからウィンドウ種類が優先ウィンドウであるものの内、スタック最下位のウィンドウを検索し、そのウィンドウが対象ウィンドウ自身であれば、何もせずS1112へ進み、そのウィンドウが対象ウィンドウ自身でなければ、そのウィンドウの直下となるよう、ウィンドウシステム12にスタック変更要求を発行し、次に、内部データベース全体に対してスタック位置情報を更新した上で、S1112に進む。

【0087】

S1105のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図12に示す。

【0088】

図12に示されるように、対象ウィンドウのみが優先ウィンドウの中のスタック最下位のウィンドウの直下に移されているのが解る。

【0089】

図11において、S1106では、対象ウィンドウと同じグループに属する通常ウィンドウを内部データベースから検索してリストアップし、S1107に進む。

【0090】

S1107では、前記リストアップした中にスタック位置確定フラグが確定済みであるウィンドウが存在するならばS1108に進み、存在しないならS1109に進む。

【0091】

S1108では、リストアップした中でスタック位置確定フラグが暫定であるウィ

ンドウがあれば、それら全てを、それらの中でみたスタック順はそのまま、リストアップした中でスタック位置確定フラグが確定済みであるウィンドウの内の最上位のウィンドウの直上に位置させ、更に対象ウィンドウをリストアップしたウィンドウの中で最下位となるよう、ウィンドウシステム 12 にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S1112に進む。

【0092】

S1108のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図 13 に示す。図 13 に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、同じグループ値を持つスタック位置確定フラグが確定済みの通常ウィンドウの直上に移され、同一グループ値を持つ通常ウィンドウ全体の中で対象ウィンドウのスタック最下位移動がなされているのが解る。

【0093】

図 11 において、S1109では、内部データベースにスタック位置確定フラグが確定済みの通常ウィンドウが存在するならばS1110に進み、存在しないならばS1111に進む。

【0094】

S1110では、S1106でリストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのまま、内部データベースの中でスタック位置確定フラグが確定済みであるスタック最上位の通常ウィンドウの直上に位置させ、更に対象ウィンドウを前記リストアップしたウィンドウの中で最下位となるよう、ウィンドウシステム 12 にスタック変更要求を発行し、次に、リストアップしたウィンドウのスタック位置確定フラグを確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S1112に進む。

【0095】

S1110のスタック変更要求によってスタックがどのように変化するかを示す具

体例となる模式図を図 14 に示す。

【0096】

図 14 に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック位置確定フラグが確定済みの通常ウィンドウ全体のスタック最上位ウィンドウの直上に移され、同一グループ値を持つウィンドウ全体の中で対象ウィンドウのスタック最上位移動がなされているのが解る。

【0097】

図 11 において、S1111では、S1106でリストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのまま、全体スタックの最下位とし、更に対象ウィンドウをリストアップしたウィンドウの中で最上位となるよう、ウィンドウシステム 12 にスタック変更要求を発行し、次に、リストアップしたウィンドウのスタック位置確定フラグを確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S1112に進む。

【0098】

S1111のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図 15 に示す。

【0099】

図 15 に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック最下位に移され、同一グループ値を持つ通常ウィンドウ全体の中で対象ウィンドウのスタック最上位移動がなされているのが解る。

【0100】

図 11 において、S1112では、処理を終了する。

【0101】

図 1 において、アプリケーションプログラム 101、102、103は、生成及びウィンドウ種類及びグループ設定した通常ウィンドウに対して、グループ単位スタック最上位移動をウィンドウマネージャ 105 に対して発行する事ができ

る。Xウィンドウシステムの場合、例えば、クライアントメッセージ機構を利用する事で容易に実現できる。今後、クライアントメッセージでウィンドウマネージャ105に対して発行するグループ単位スタック最上位移動要求のメッセージの事をグループ単位スタック最上位移動要求メッセージと呼ぶ事とする。そして、このグループ単位スタック最上位移動要求メッセージには、移動したいグループ値が含まれているものとする。

【0102】

アプリケーションプログラムがグループ単位スタック最上位移動要求メッセージ (ClientMessage) を発行した場合、ウィンドウシステム12経由でウィンドウマネージャ105に転送される。グループ単位スタック最上位移動要求メッセージを受け取った際のウィンドウマネージャ105の動作例となるフローを図16に示す。

【0103】

図16に示されるように、S1601では、該時点で受け取ったグループ単位スタック最上位移動要求メッセージ (ClientMessage) から対象グループ値を獲得して、S1602に進む。

【0104】

S1602では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム12に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値とプロパティ獲得済みである事を内部データベースに記憶し、S1603に進む。

【0105】

S1603では、対象グループ値を持つ通常ウィンドウ情報を内部データベースから検索してリストアップし、S1604に進む。

【0106】

S1604では、内部データベースにスタック位置確定情報が確定済み且つグループ値情報が対象グループ値とは異なる通常ウィンドウが存在すればS1605に進み、存在しなければS1606に進む。

【0107】

S1605では、リストアップした中でのスタック順はそのまま、内部データベースの中でスタック位置確定フラグが確定済み且つグループ値情報が対象グループ値とは異なる通常ウィンドウの内でスタック最上位のウィンドウの直上となるよう、ウィンドウシステム12にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S1607に進む。

【0108】

S1605のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図17に示す。

【0109】

図17に示されるように、対象グループ値を持つ通常ウィンドウが、スタック位置確定フラグが確定済み且つグループ値情報が対象グループ値とは異なる通常ウィンドウ全体のスタック最上位ウィンドウの直上に移されているのが解る。

【0110】

図16におけるS1606では、リストアップした中でのスタック順はそのまま、全体スタックの最下位となるよう、ウィンドウシステム12にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S1607に進む。

S1606のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図18に示す。

【0111】

図18に示されるように、対象グループ値を持つ通常ウィンドウが、全体スタックの最下位に移されているのが解る。

【0112】

図16において、S1607では、処理を終了する。

【0113】

図1において、アプリケーションプログラム101、102、103は、生成及びウィンドウ種類及びグループ設定した通常ウィンドウに対して、グループ単位スタック最下位移動をウィンドウマネージャ105に対して発行する事ができる。Xウィンドウシステムの場合、例えば、クライアントメッセージ機構を利用する事で容易に実現できる。今後、クライアントメッセージでウィンドウマネージャ105に対して発行するグループ単位スタック最下位移動要求のメッセージの事をグループ単位スタック最下位移動要求メッセージと呼ぶ事とする。そして、このグループ単位スタック最下位移動要求メッセージには、移動したいグループ値が含まれているものとする。

【0114】

アプリケーションプログラムがグループ単位スタック最下位移動要求メッセージ (ClientMessage) を発行した場合、ウィンドウシステム12経由でウィンドウマネージャ105に転送される。

【0115】

グループ単位スタック最下位移動要求メッセージを受け取った際のウィンドウマネージャ105の動作例となるフローを図19に示す。

【0116】

図19に示されるように、S1901では、該時点で受け取ったグループ単位スタック最下位移動要求メッセージ (ClientMessage) から対象グループ値を獲得して、S1902に進む。

【0117】

S1902では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム12に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値とプロパティ獲得済みである事を内部データベースに記憶し、S1903に進む。

【0118】

S1903では、対象グループ値を持つ通常ウィンドウを内部データベースから検索してリストアップし、S1904に進む。

【0119】

S1904では、リストアップした中でのスタック順はそのまま、全体スタックの最下位となるよう、ウィンドウシステム12にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、S1905に進む。

【0120】

S1904のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図20に示す。

【0121】

図20に示されるように、対象グループ値を持つ通常ウィンドウが、全体スタックの最下位に移されているのが解る。

【0122】

図19におけるS1905では、処理を終了する。

【0123】

図1において、アプリケーションプログラム101、102、103は、生成したウィンドウに対して破棄要求を発行する。

【0124】

ウィンドウシステム12は、破棄要求を受け取ると、該当するウィンドウを管理から外して削除する。そして、ウィンドウマネージャ105にウィンドウ破棄通知イベント (DestroyNotify) を発行する。

【0125】

ウィンドウマネージャ105は、ウィンドウ破棄通知イベント (DestroyNotify) を受け取ると、そこからウィンドウ識別子を抽出して、該当するウィンドウ情報を内部データベースから検索し削除した後、内部データベース全体に対して必要に応じてスタック位置を更新し、処理を終了する。

【0126】

以上のことにより、各通常ウィンドウはそのウィンドウの表示までに、ウィンドウのスタック順がグループ毎に纏められるので、グループ最上位のウィンドウ

は、他のグループのウィンドウの下に重ねられて表示される事が無くなる。また、ウィンドウ管理プログラムが、スタック変更要求を受け取る事を切っ掛けにして、スタック変更対象となるグループに属する通常ウィンドウ全てを、その中のスタック順は保持したままで纏めた上でスタック変更を実施するので、アプリケーションプログラムは、自グループのウィンドウのスタック順を把握し易くなる。また、ウィンドウ管理プログラムにグループ内のスタック最上位移動とグループ内のスタック最下位移動の機能を設ける事で、グループ内で通常ウィンドウのスタック順を操作できるので、各アプリケーションプログラムは、他のアプリケーションプログラムの通常ウィンドウ表示を気にする事無く、スタック変更を行う事ができる。また、ウィンドウ管理プログラムにグループ単位のスタック最上位移動と最下位移動の機能を設ける事で、例えば、グループをアプリケーションプログラム単位で振り分けた場合だと、そのアプリケーションに属するウィンドウのメンバーを知らなくても、スタック最前面にそのアプリケーションに属する全通常ウィンドウを移動させる事が可能となる。また、優先ウィンドウを設ける事で、グループ単位のスタック順に関係無く、最優先で表示するウィンドウのサービスを提供することができる。

【0127】

なお、本実施の形態では、ウィンドウマネージャのウィンドウのウィンドウ種類及びグループ値取得のタイミングをマップ要求イベント (MapRequest) 受信時及びスタック変更要求受信時としたが、これに限定されるものではなく、アプリケーションプログラムがウィンドウ種類及びグループ設定をしたタイミングで取得して内部データベースに記憶しておいても良く、その方がむしろ処理効率が良い。更に、このタイミングで、スタック順をグループ毎に纏めても構わない。また、ウィンドウ種類及びグループ値の変更を認めても良いし、認めなくても良い。認める場合は、ウィンドウ種類及びグループ設定のタイミングで、ウィンドウ種類が優先ウィンドウから通常ウィンドウに変更された場合は、スタック位置確定フラグを暫定に変更する。逆にウィンドウ種類が通常ウィンドウから優先ウィンドウに変更された場合は、該時点のスタック最上位に該ウィンドウを位置させるようにスタック変更を行う。また、ウィンドウ種類に変更が無く、グループ値が

変更となった場合は、スタック位置確定フラグを暫定に変更する。そして、内部データベースを更新すれば良い。また、認めない場合は、ウィンドウ種類及びグループ設定のタイミングで、プロパティ獲得フラグをチェックして、獲得済みであれば、該設定を無視すれば良い。

【0128】

また、ウィンドウ種類として優先ウィンドウと通常ウィンドウの2種類を制御する例を示したが、通常ウィンドウの1種類だけであっても容易に実施できる事は言うまでも無い。

【0129】

また、ウィンドウマネージャがルートウィンドウの子ウィンドウであるトップレベルウィンドウのスタック順を制御する場合を例に示したが、トップレベルウィンドウの子孫ウィンドウに対しても実施する事ができる事は言うまでも無い。

【0130】

また、ウィンドウ管理プログラムをXウィンドウシステムとウィンドウマネージャとして、アプリケーションプログラムとのインターフェースにウィンドウ生成要求、マップ要求を用いたが、これに限定されるものではなく、表示要求に相当するインターフェースさえあれば良く、その場合、ウィンドウ管理プログラムは、ウィンドウ表示要求受信時に、本実施の形態で示したウィンドウ生成要求受信時の処理とマップ要求受信時の処理を纏めて行えば良い。

【0131】

また、ウィンドウのスタック順をグループ毎に一連とする際、単にウィンドウのスタック順を変更するだけであつたが、例えば、ウィンドウ管理プログラムが、グループ毎に代表ウィンドウを1枚生成し、ウィンドウのスタック順をグループ毎に一連とする際に、前記ウィンドウを前記代表ウィンドウの子ウィンドウとするようにして制御することも容易である。そうすることによって、グループ単位のスタック移動やグループ単位のウィンドウの表示/非表示の切り替えを代表ウィンドウに対する制御で実施できる分、グループ内のウィンドウが多いほど、処理負荷を軽減することができる。代表ウィンドウ使用下でのスタックの様子を示す例となる模式図を図25に示す。

【0132】

また、最上位グループよりスタック下位のウィンドウ表示について、ウィンドウ管理プログラムでは非表示とするような制御はしていないが、非表示とする制御を行う事も容易に実現できる。その場合、ウィンドウ情報に表示/非表示を示すマップフラグを追加して、アプリケーションプログラムの表示要求状態を記憶しておき、該ウィンドウの所属するグループが通常ウィンドウの中でスタック最上位となった時、前記マップフラグをチェックし、表示となっていればマップ処理を行い、非表示となっていればアンマップのままとする。逆に、該ウィンドウの所属するグループが通常ウィンドウの中でスタック最上位でなくなった際には、前記マップフラグが表示となっていればアンマップ処理を行い、非表示となっていれば何もしない。これにより、通常ウィンドウの中の最上位グループよりスタック下位のグループが何を表示しようとしても、表示装置に表示されないのので、最上位グループのウィンドウで覆われない部分に中途半端に映り込む心配が無い。

【0133】

また、通常ウィンドウの最上位グループに属する全ウィンドウの中のスタック最下位のウィンドウの直下に、ウィンドウ管理プログラムが生成及びマップする特定のウィンドウを位置させなかったが、位置させることも容易に実現できる。これにより、例えば、前記特定のウィンドウを表示装置のスクリーン一杯に表示しておけば、前記特定のウィンドウのスタック下位のグループが何を表示しようとしても、表示装置に表示されないのので、最上位グループのウィンドウで覆われない部分に中途半端に映り込む心配が無い。前記特定のウィンドウを使用した場合のスタックの様子を示す例となる模式図を図26に示し、また、スクリーンに表示されている例となる模式図を図27に示す。

【0134】

また、アプリケーションプログラムが3つの場合を例に示したが、これに限定されるものではなく、1つ以上であれば良い。

【0135】

また、各アプリケーションプログラムに1つのグループを割り当てた場合を例

に示したが、これに限定されるものではなく、1つのアプリケーションに複数のグループを割り当てても良く、また、複数のアプリケーションに1つのグループを割り当てても構わない。

【0136】

また、ウィンドウ管理プログラムをXウィンドウシステムとウィンドウマネージャとして、両者間でイベントや各種要求を送受信し合う例を示したが、これに限定されるものではなく、Xウィンドウシステムである必要は無いし、また、2つに分けてイベントや各種要求を送受信する構成である必要もなく、一体化して実施する場合は、イベントやメッセージのインターフェースを関数ベースに変える事で本実施の形態を容易に応用できる。

【0137】

また、通常ウィンドウの最上位グループに属する全ウィンドウの中のスタック最上位のウィンドウの直上に、ウィンドウ管理プログラムが生成する特定のウィンドウを位置させなかったが、位置させることも容易に実現できる。これにより、最上位グループを別のグループに入れ替える際、Xウィンドウシステムの場合、XRestackWindows (R) 関数を使用する事となるが、ウィンドウ群のスタック移動先を示すために使用できる。もし、最上位グループのウィンドウよりスタック上位にウィンドウが存在しなかった場合、存在する全てのウィンドウの並び順を決め、前記XRestackWindows (R) の引数に代入せねばならないが、前記特定のウィンドウがあれば、前記特定のウィンドウと前記ウィンドウ群を引数に代入するだけで済む。

【0138】

また、Xウィンドウシステムを用いる場合、ウィンドウマネージャは、スタックを操作するためにXRestackWindows (R) 関数を使用する。XRestackWindows (R) 関数の引数には、ウィンドウ群とそのウィンドウ群の移動先を示すウィンドウを指定するが、それらの何れかが破棄されていた場合、スタック操作が意図した通りに実施されない事がある。そのための対策として、例えば、ウィンドウマネージャは、ウィンドウ破棄通知イベント (DestroyNotify) を受け取った際に、該当するウィンドウ情報を内部データベースから検索し削除し、内部データベ

ース全体に対して必要に応じてスタック位置を更新した後、次に、ウィンドウシステムの該時点でのスタックツリーを問い合わせで獲得し、ウィンドウマネージャが認識していない新規のウィンドウは対象から外し、ウィンドウマネージャのスタックツリーとの整合性をチェックし、ウィンドウマネージャの認識と異なっている場合は、ウィンドウマネージャのスタックツリーに合わせ込むスタック変更要求を発行し、認識が合うまで、ウィンドウシステムの該時点でのスタックツリー問い合わせ、整合性チェック、スタック変更要求の発行を繰り返すようにしても良い。これにより、ウィンドウシステムがスタック変更処理に失敗しても、ウィンドウシステム間とのスタック認識の整合性を確保でき、処理が破綻する事がなくなる。

【0139】

また、ウィンドウマネージャが、XRestackWindows (R) を実行した際にフラグを立て、ウィンドウ破棄通知イベント (DestroyNotify) を受け取った際に、前記フラグが立っている場合にのみ、前記フラグを下げると共にスタックツリーの整合性のための処理を行うようにしても良い。これにより、スタックツリーの整合性のための処理の実行頻度を軽減できる。

【0140】

また、コンピュータ 1 とは、家庭用コンピュータや携帯電話、PDA (Personal Digital Assistance)、セットトップボックス、デジタルスチルカメラ、カメラ一体型VTR等、プログラムが動作するためのCPUとメモリを実装して表示装置に複数のウィンドウを表示しようとする装置であれば何でも良い事は言うまでもない。

【0141】

また、表示装置 11 をコンピュータ 1 の一部としたが、物理上で一体化している事を指すものではなく、コンピュータ 1 と有線/無線を問わず、何らかの形で接続されていさえすれば良い。

【0142】

また、本実施の形態のウィンドウスタック制御方法を内蔵するプログラムは、ROMやディスク等に記憶して供給する事ができるほか、ネットワークを介して供

給する事が可能である事は言うまでもない。

【0143】

(実施の形態2)

本実施の形態のウィンドウスタック制御方法は、実施の形態1を変形したものであり、実施の形態1と実施の形態2の違いは、マップ要求イベント(MapRequest)を受け取った際のウィンドウマネージャ105の動作と、アプリケーションプログラムがウィンドウ種類及びグループ値を設定するタイミングでウィンドウマネージャ105はスタック順をグループ毎に纏めてはいけないことだけである。よって、本実施例では、ウィンドウマネージャ105のマップ要求イベント(MapRequest)を受け取った際のウィンドウマネージャ105の動作についてのみ説明する。

【0144】

マップ要求イベント(MapRequest)を受け取った際のウィンドウマネージャ105の動作例となるフロー図を図21に示す。

【0145】

図21に示されるように、S2101では、該時点で受け取ったマップ要求イベント(MapRequest)からマップ対象ウィンドウの識別子を獲得して、S2102に進む。

【0146】

S2102では、マップ対象ウィンドウの識別子で内部データベースを検索し、対応するウィンドウ情報が存在する場合はS2103へ進み、存在しない場合はS2115へ進む。

【0147】

S2103では、内部データベースで対象ウィンドウのスタック位置確定フラグが確定済みかどうかをチェックし、確定済みの場合はS2114に進み、暫定の場合はS2104に進む。

【0148】

S2104では、対象ウィンドウのプロパティ獲得フラグをチェックし、獲得済みの場合はS2106に進み、未獲得の場合はS2105に進む。

【0149】

S2105では、対象ウィンドウのウィンドウ種類及びグループ値をウィンドウシステムに問い合わせて獲得し、内部データベースに格納し、S2106に進む。

【0150】

S2106では、対象ウィンドウのウィンドウ種類をチェックし、優先ウィンドウの場合は、S2114に進み、通常ウィンドウの場合は、S2107に進む。

【0151】

S2107では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム12に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値とプロパティ獲得済みである事を内部データベースに記憶し、S2108に進む。

【0152】

S2108では、対象ウィンドウと同じグループに属する通常ウィンドウを内部データベースから検索してリストアップし、S2109に進む。 S2109では、前記リストアップした中にスタック位置確定フラグが確定済みであるウィンドウが存在するならばS2110に進み、存在しないならS2111に進む。

【0153】

S2110では、リストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのまま、リストアップした中でスタック位置確定フラグが確定済みであるウィンドウの内の最上位のウィンドウの直上となるよう、ウィンドウシステム12にスタック変更要求を発行し、S2114に進む。 S2110のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図22に示す。

【0154】

図22に示されるように、マップ対象ウィンドウ及びマップ対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、前記グループ値を持つスタック位置確定フラグが確定済みの通常ウィンドウの直上に移されているのが解かる。

【0155】

図 2 1 において、S2111では、内部データベースにスタック位置確定フラグが確定済みの通常ウィンドウが存在するならばS2112に進み、存在しないならばS2113に進む。

【0156】

S2112では、S2108でリストアップした中でスタック位置確定フラグが暫定である通常ウィンドウ全てを、それらの中でみたスタック順はそのまま、内部データベースの中でスタック位置確定フラグが確定済みであるスタック最上位の通常ウィンドウの直上となるよう、ウィンドウシステム 1 2 にスタック変更要求を発行し、S2114に進む。

【0157】

S2112のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図 2 3 に示す。 図 2 3 に示されるように、マップ対象ウィンドウ及びマップ対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック位置確定フラグが確定済みの通常ウィンドウ全体のスタック最上位ウィンドウの直上に移されているのが解る。

【0158】

図 2 1 において、S2113では、S2108でリストアップした中でスタック位置確定フラグが暫定である通常ウィンドウ全てを、それらの中でみたスタック順はそのまま、全体スタックの最下位となるよう、ウィンドウシステム 1 2 にスタック変更要求を発行し、S2114に進む。

【0159】

S2113のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図 2 4 に示す。

【0160】

図 2 4 に示されるように、マップ対象ウィンドウ及びマップ対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック最下位に移されているのが解る。

【0161】

図 2 1 において、S2114では、ウィンドウシステム 1 2 に対してマップ対象ウ

ィンドウのマップ要求を発行し、内部データベースの対象ウィンドウのウィンドウ情報の内、スタック位置確定フラグが未確定を示している場合は確定済みに更新し、また、内部データベース全体に対して必要に応じてスタック位置を更新した上で、S2115に進む。

【0162】

S2115では、処理を終了する。

【0163】

以上のことにより、ウィンドウ管理プログラムが、マップ対象ウィンドウと同じグループ値を持つスタック位置が暫定の通常ウィンドウ群を、その通常ウィンドウ群の中でのスタック上下関係は保持しつつスタック位置を移してグループ毎に纏めるため、マップによって、同一グループ内でみたスタック順は変更されず、アプリケーションの同一グループ内のスタック順管理の負担が軽減される。更に、実施の形態1で示したように、スタック変更要求受信時にも、対象ウィンドウと同じグループ値を持つスタック位置が暫定の通常ウィンドウ群を、その通常ウィンドウ群の中でのスタック上下関係は保持しつつグループ毎に纏める事で、アプリケーションプログラムは、自グループに属するウィンドウのスタック順をウィンドウの生成順を基本として管理する事ができる。これは、Xウィンドウシステムの基本と一致しており、アプリケーションプログラムは、自グループの通常ウィンドウについて、一般的なスタック順決定ルールである生成順ベースで重なりを管理すれば良くなる。

【0164】

なお、本実施の形態では、マップ要求イベント(MapRequest)受信時に、ウィンドウマネージャのウィンドウのグループ値取得を行っているが、これに限定されるものではなく、実施の形態1と同様、アプリケーションプログラムがウィンドウ種類及びグループ設定をしたタイミングでグループ値を取得して内部データベースにウィンドウ種類とグループ値とプロパティ獲得済みである事を記憶しておいても良く、その方がむしろ処理効率が良い。しかし、実施の形態1と違い、前記タイミングで、スタック順をグループ毎に纏めてはいけない。

【0165】

【発明の効果】

以上のように本発明によれば、ウィンドウ制御方法において、アプリケーションプログラムがウィンドウにグループを指定し、ウィンドウ管理プログラムがウィンドウのスタック順をグループ毎に一連とすることにより、グループ最上位のウィンドウは、他のグループのウィンドウの下に重ねられる事が無くなり、最前面でウィンドウを表示中のアプリケーションのウィンドウ表示に、他のアプリケーションが不用意に影響を及ぼす事を無くす事ができる。特に、画面が小さく、結果として、ウィンドウの重なり易い携帯電話やPDA等で、その効果は顕著である。

【図面の簡単な説明】**【図 1】**

本発明の実施の形態 1 のウィンドウスタック制御方法を実現するシステムの構成例を示す図

【図 2】

本発明実施の形態 1 のマップ要求イベント (MapRequest) を受け取った際のウィンドウマネージャの動作例となるフロー図

【図 3】

本発明の実施の形態 1 の図 2 の S208 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 4】

本発明の実施の形態 1 の図 2 の S210 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 5】

本発明の実施の形態 1 の図 2 の S211 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 6】

本発明の実施の形態 1 のスタック最上位移動要求イベントを受け取った際のウィンドウマネージャの動作例となるフロー図

【図 7】

本発明の実施の形態 1 の図 6 の S605 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 8】

本発明の実施の形態 1 の図 6 の S608 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 9】

本発明の実施の形態 1 の図 6 の S610 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 10】

本発明の実施の形態 1 の図 6 の S611 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 11】

本発明の実施の形態 1 のスタック最下位移動要求イベントを受け取った際のウィンドウマネージャの動作例となるフロー図

【図 12】

本発明の実施の形態 1 の図 11 の S1105 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 13】

本発明の実施の形態 1 の図 11 の S1108 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 14】

本発明の実施の形態 1 の図 11 の S1110 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 15】

本発明の実施の形態 1 の図 11 の S1111 のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 16】

本発明の実施の形態 1 のグループ単位スタック最上位移動要求メッセージを受け取った際のウィンドウマネージャの動作例となるフロー図

【図 17】

本発明の実施の形態 1 の図 16 のS1605のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 18】

本発明の実施の形態 1 の図 16 のS1606のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 19】

本発明の実施の形態 1 のグループ単位スタック最下位移動要求メッセージを受け取った際のウィンドウマネージャの動作例となるフロー図

【図 20】

本発明の実施の形態 1 の図 19 のS1904のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 21】

本発明の実施の形態 2 のマップ要求イベント (MapRequest) を受け取った際のウィンドウマネージャの動作例となるフロー図

【図 22】

本発明の実施の形態 2 の図 21 のS2110のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 23】

本発明の実施の形態 2 の図 21 のS2112のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 24】

本発明の実施の形態 2 の図 21 のS2113のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図

【図 25】

本発明の実施の形態 1 の代表ウィンドウ使用下でのスタックの様子を示す例となる模式図

【図 26】

本発明の実施の形態 1 の最上位グループに属する全ウィンドウの中のスタック

最下位のウィンドウの直下に、ウィンドウ管理プログラムが生成及びマップする特定のウィンドウを位置させて使用した場合のスタックの様子を示す例となる模式図

【図 27】

本発明の実施の形態 1 の最上位グループに属する全ウィンドウの中のスタック最下位のウィンドウの直下に、ウィンドウ管理プログラムが生成及びマップする特定のウィンドウを位置させて使用した場合のスクリーンに表示されている例となる模式図

【図 28】

従来例におけるコンピュータ上で動作する 3 種類のアプリケーションが合計 5 枚のウィンドウを表示装置であるスクリーンに表示する際の表示例となる模式図

【図 29】

従来例におけるウィンドウスタック順の例となる模式図

【図 30】

従来例におけるウィンドウスタック順の例となる模式図

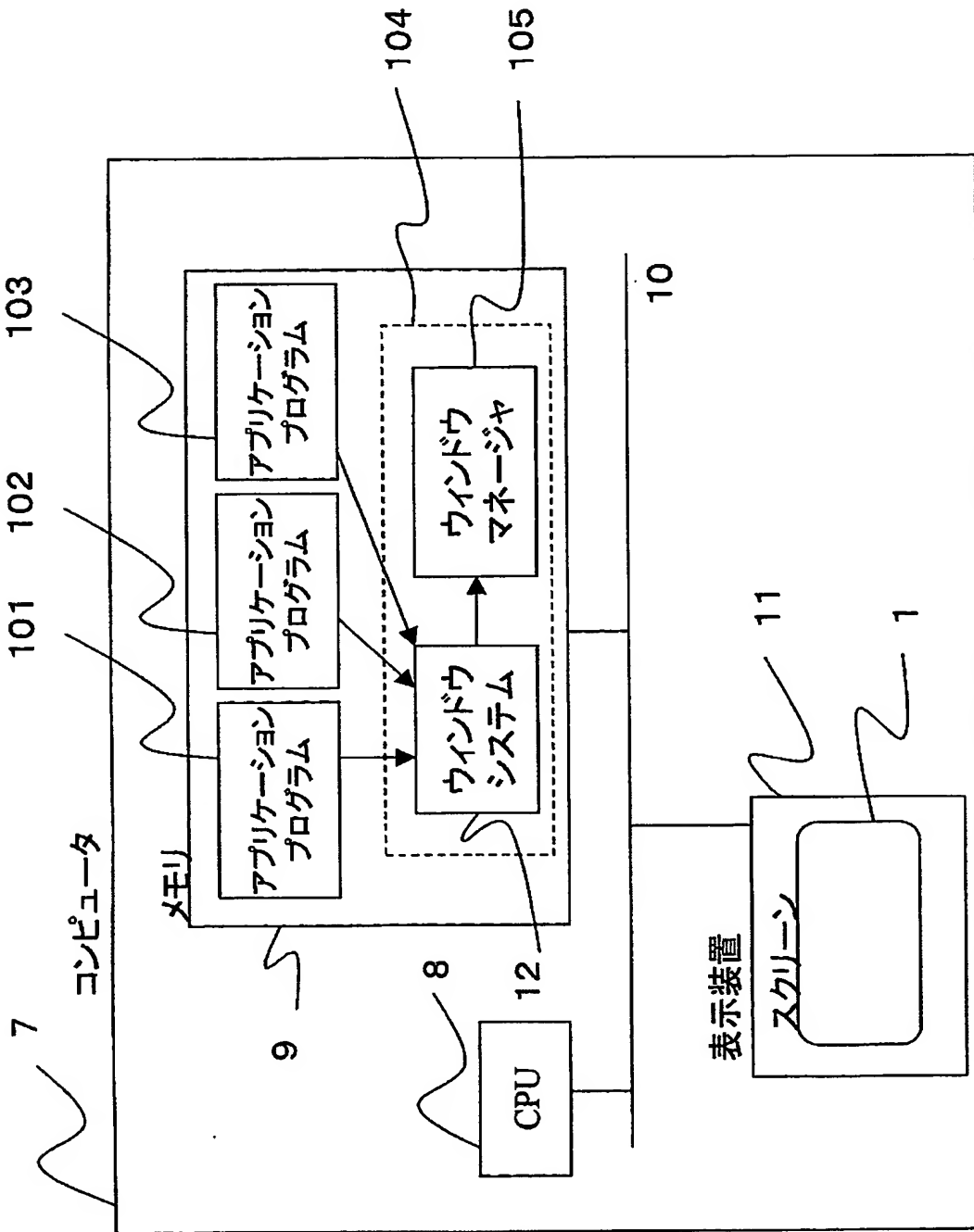
【符号の説明】

- 1 スクリーン
- 7 コンピュータ
- 8 CPU (Central Processing Unit)
- 9 メモリ
- 10 バス
- 11 表示装置
- 101、102、103 アプリケーションプログラム
- 104 ウィンドウ管理プログラム
- 105 ウィンドウマネージャ

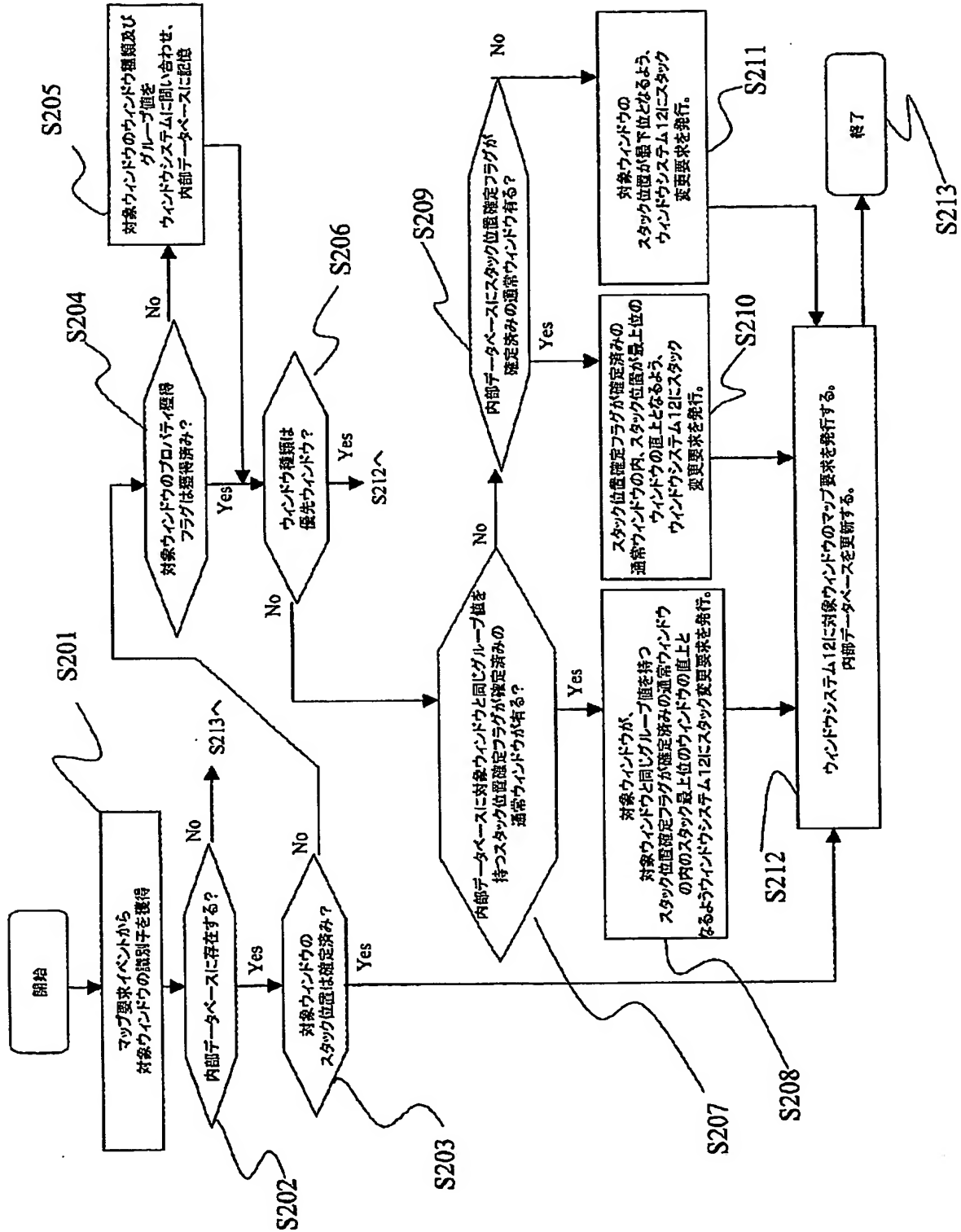
【書類名】

図面

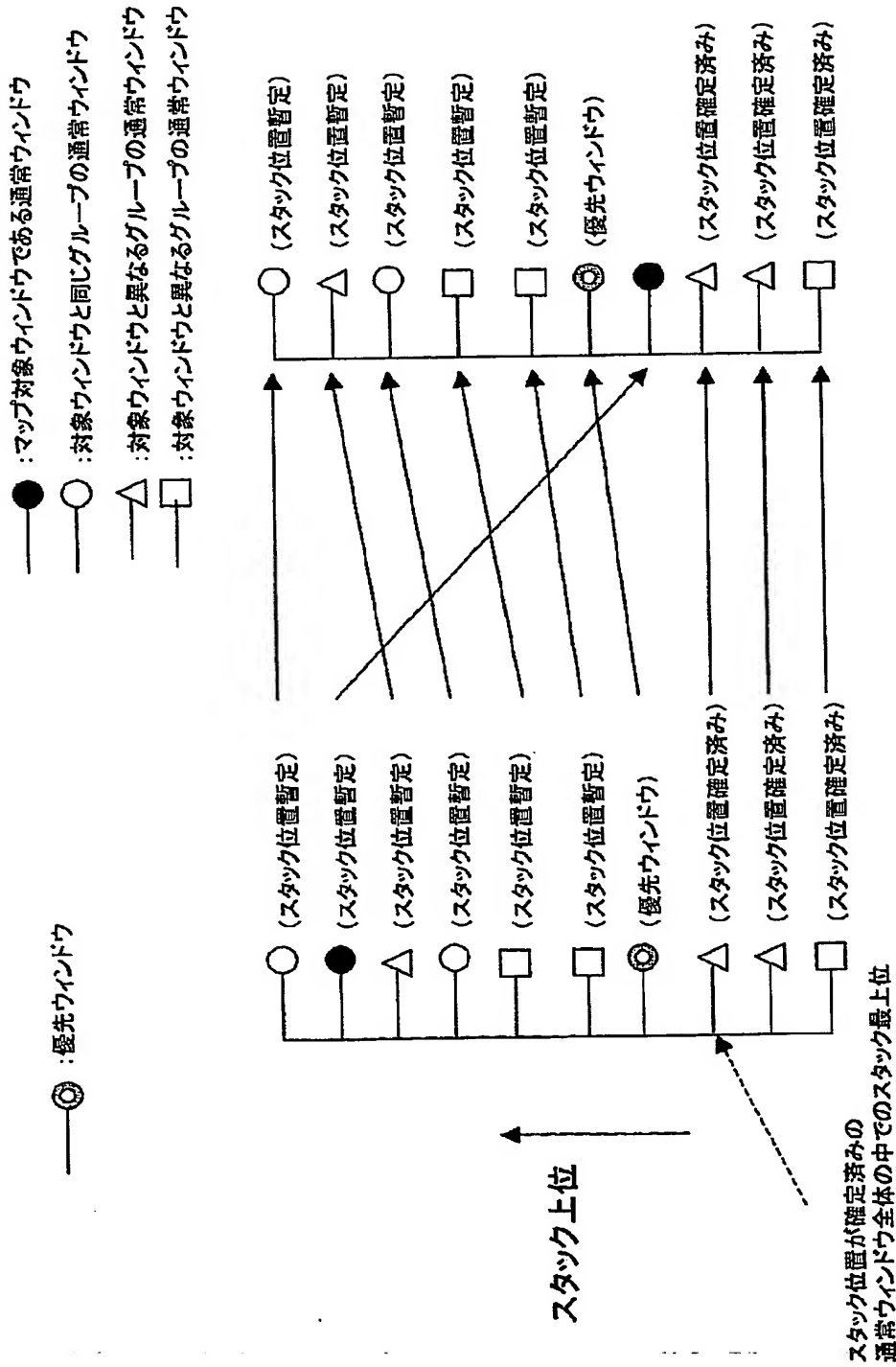
【図 1】



【図2】



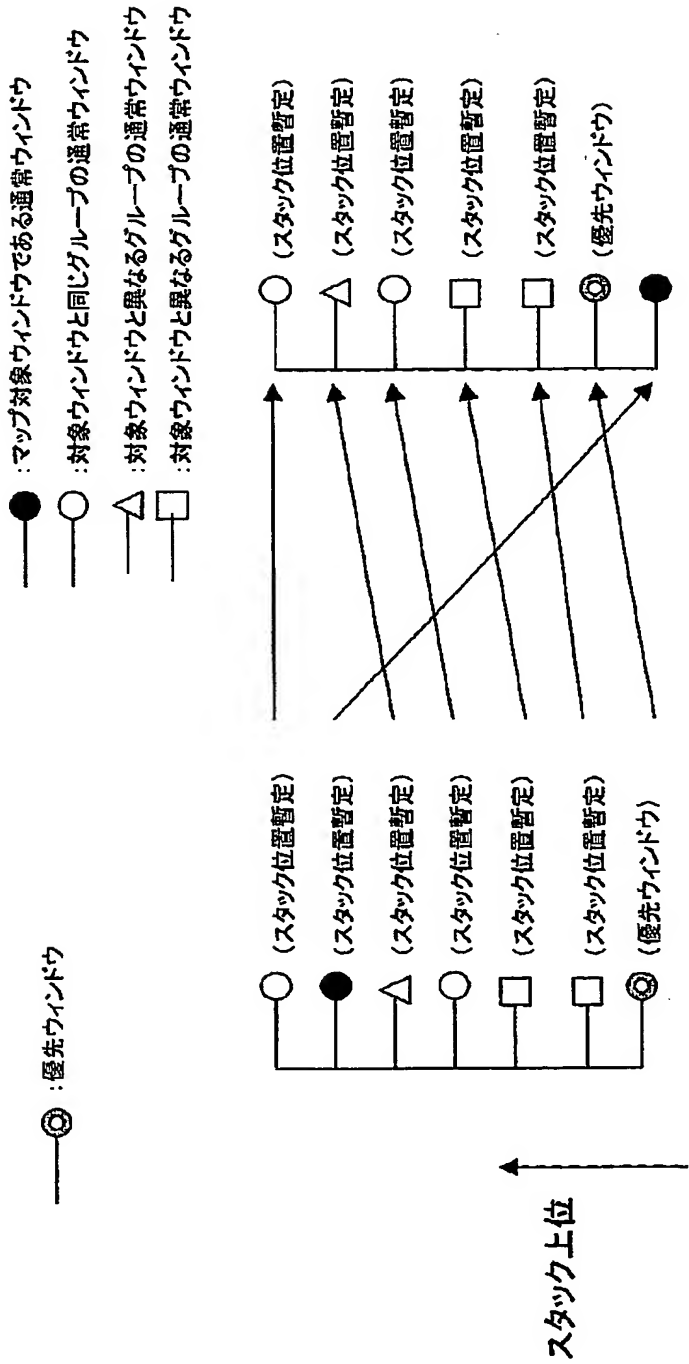
【図 4】



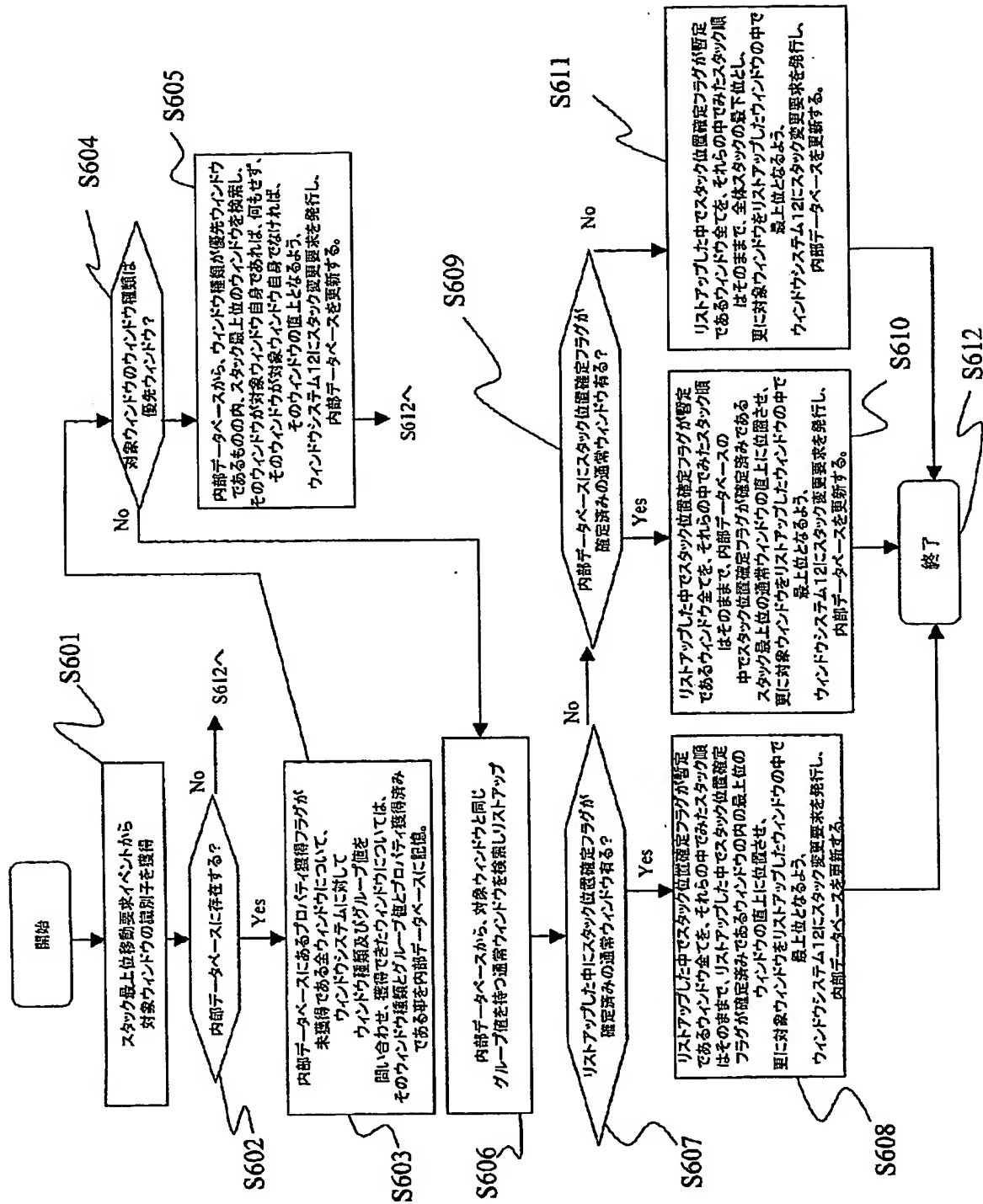
スタック変更後

スタック変更前

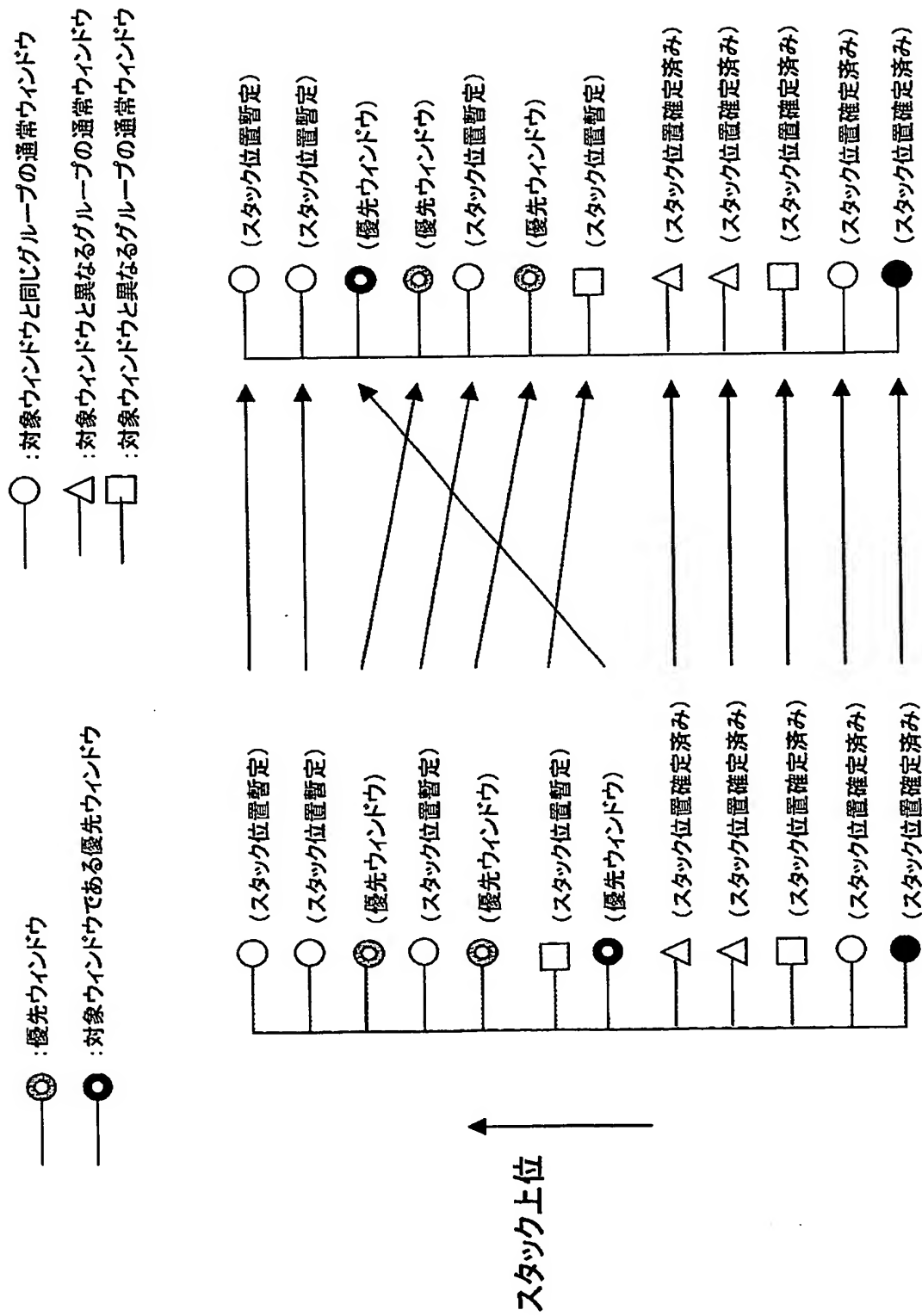
【図 5】



【図 6】



【図 7】



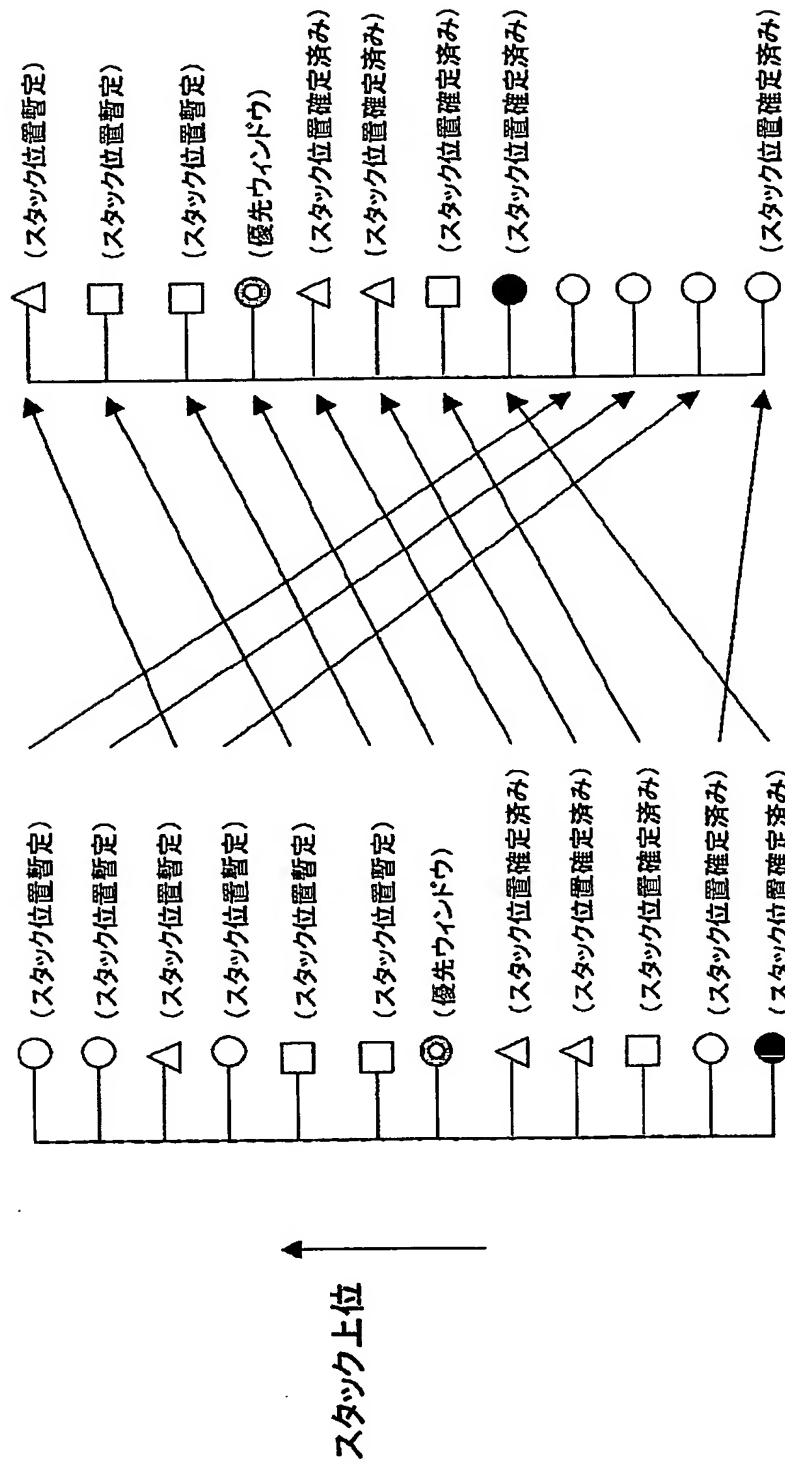
スタック変更後

スタック変更前

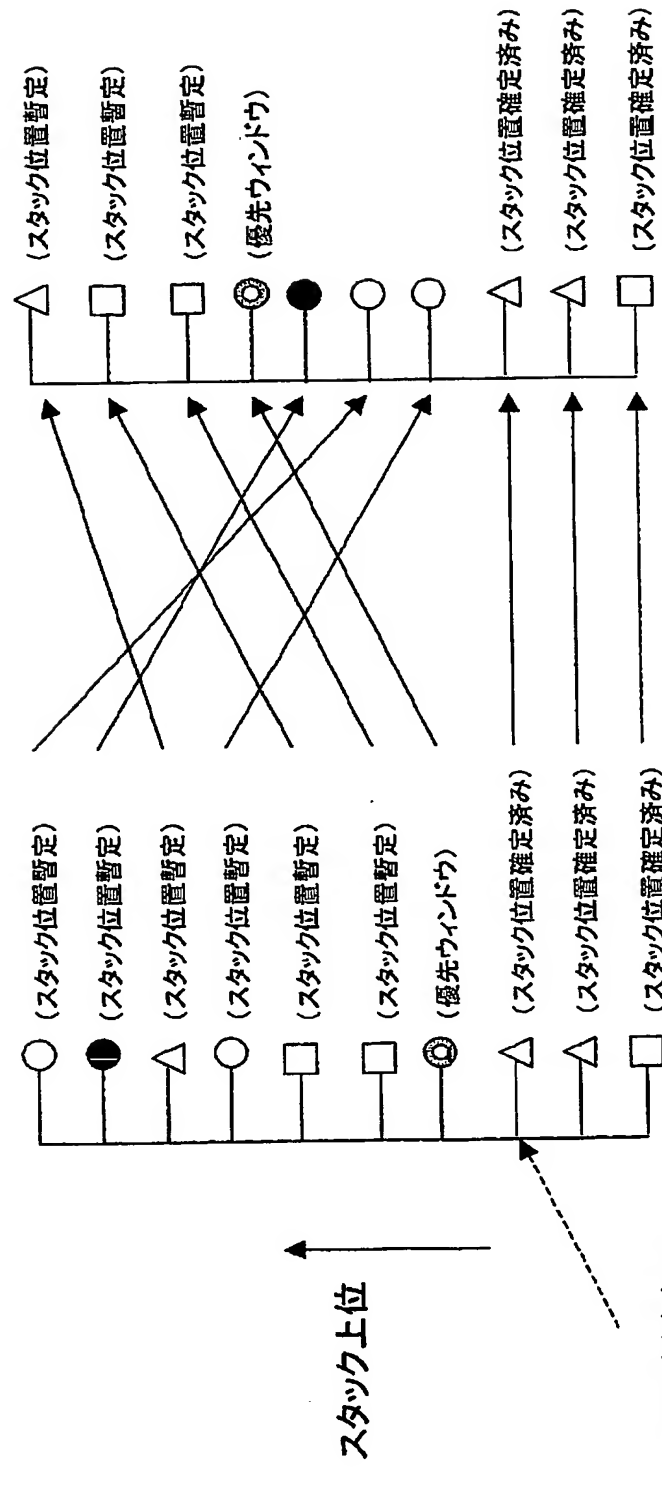
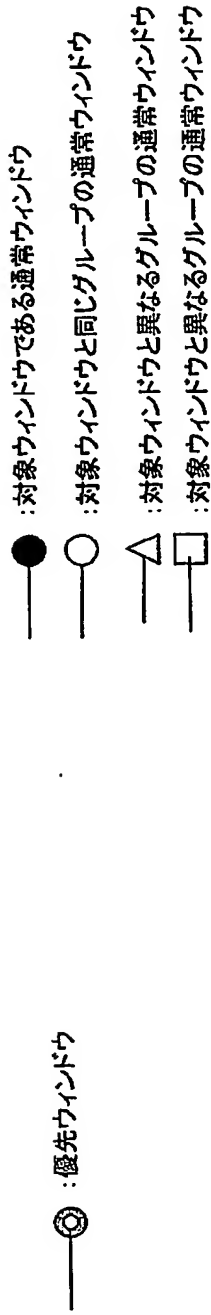
【図 8】

- : 対象ウィンドウである通常ウィンドウ
- : 対象ウィンドウと同じグループの通常ウィンドウ
- △ : 対象ウィンドウと異なるグループの通常ウィンドウ
- : 対象ウィンドウと異なるグループの通常ウィンドウ

◎ : 優先ウィンドウ



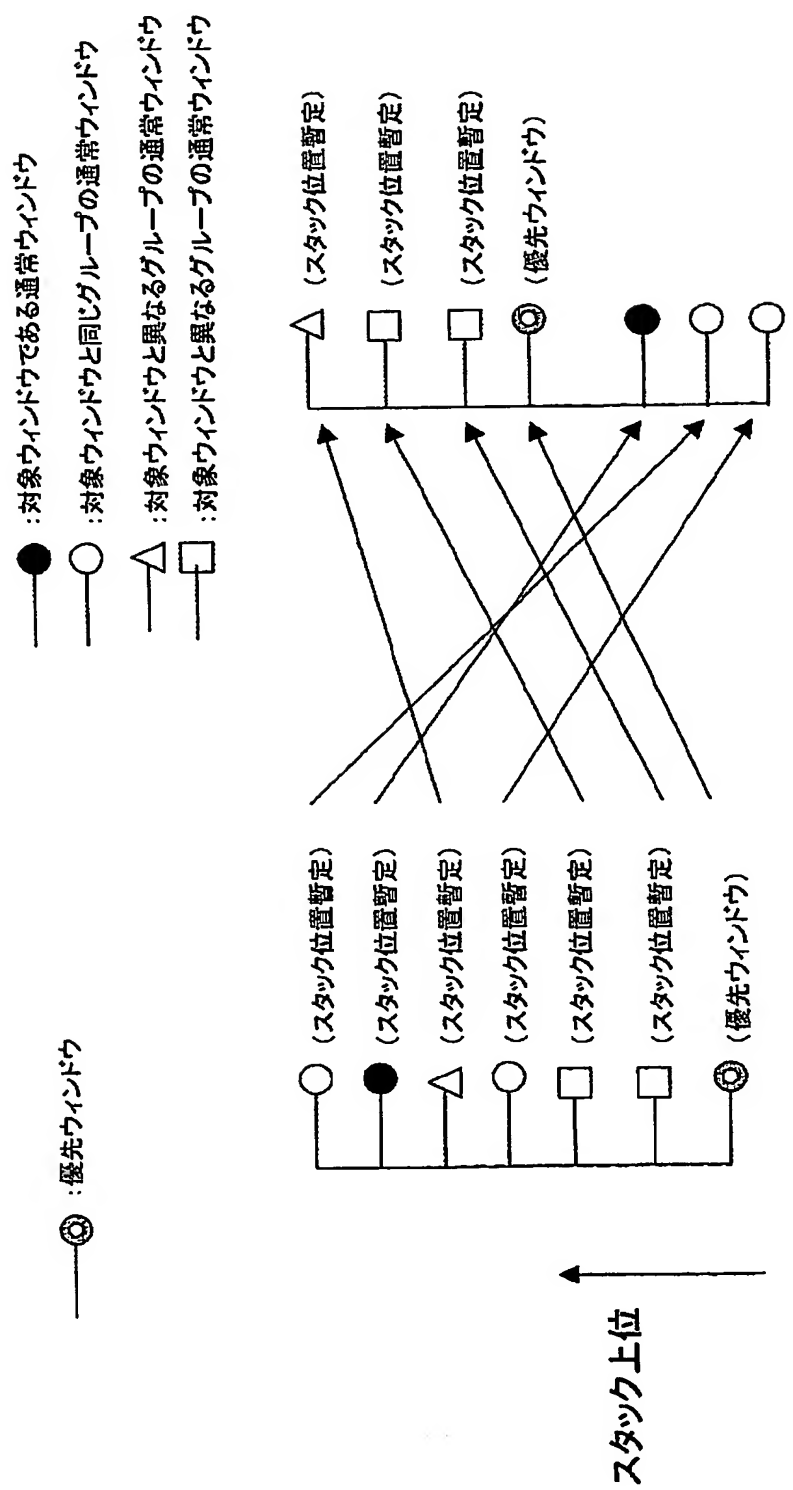
【図 9】



スタック変更後

スタック変更前

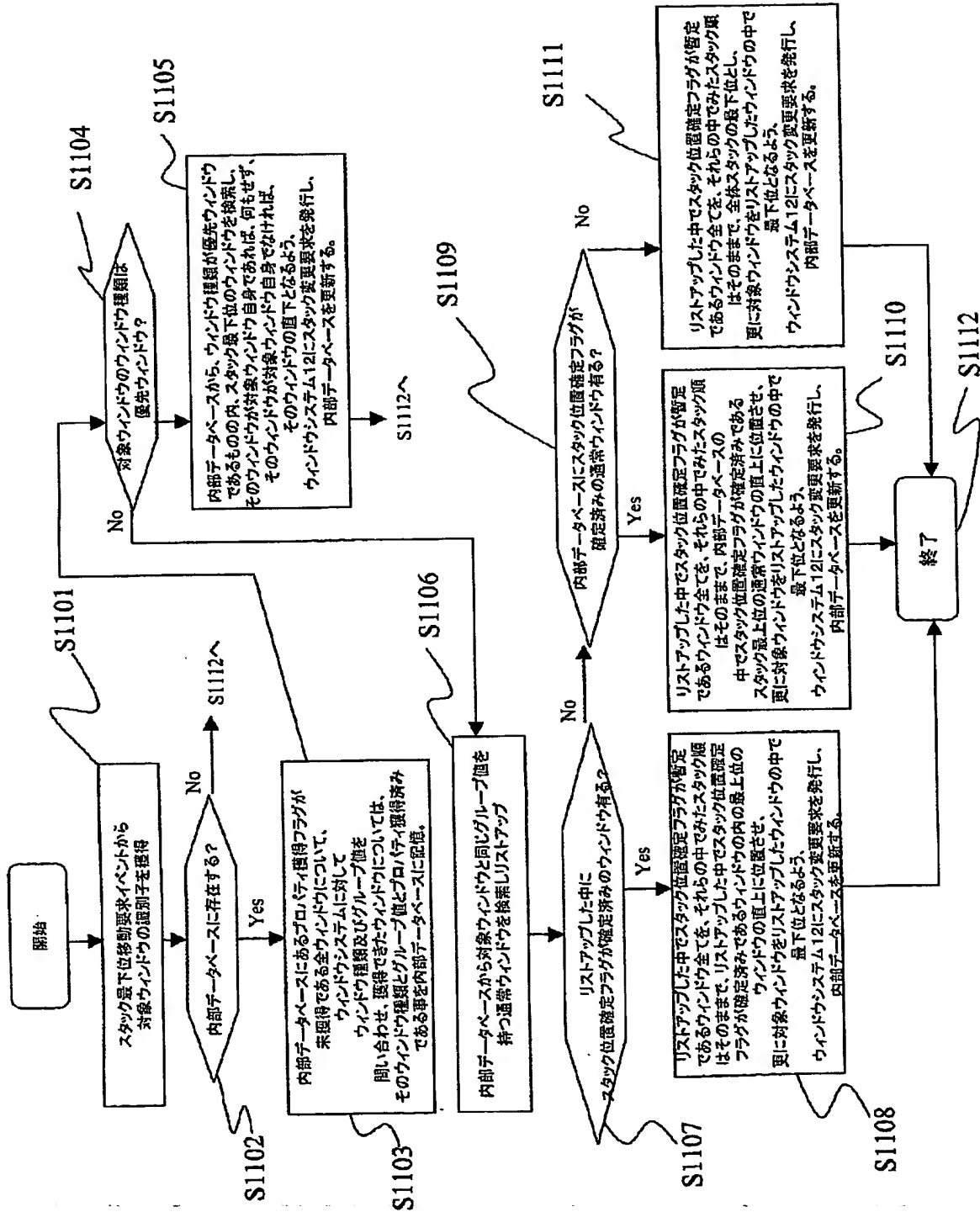
【図 10】



スタック変更後

スタック変更前

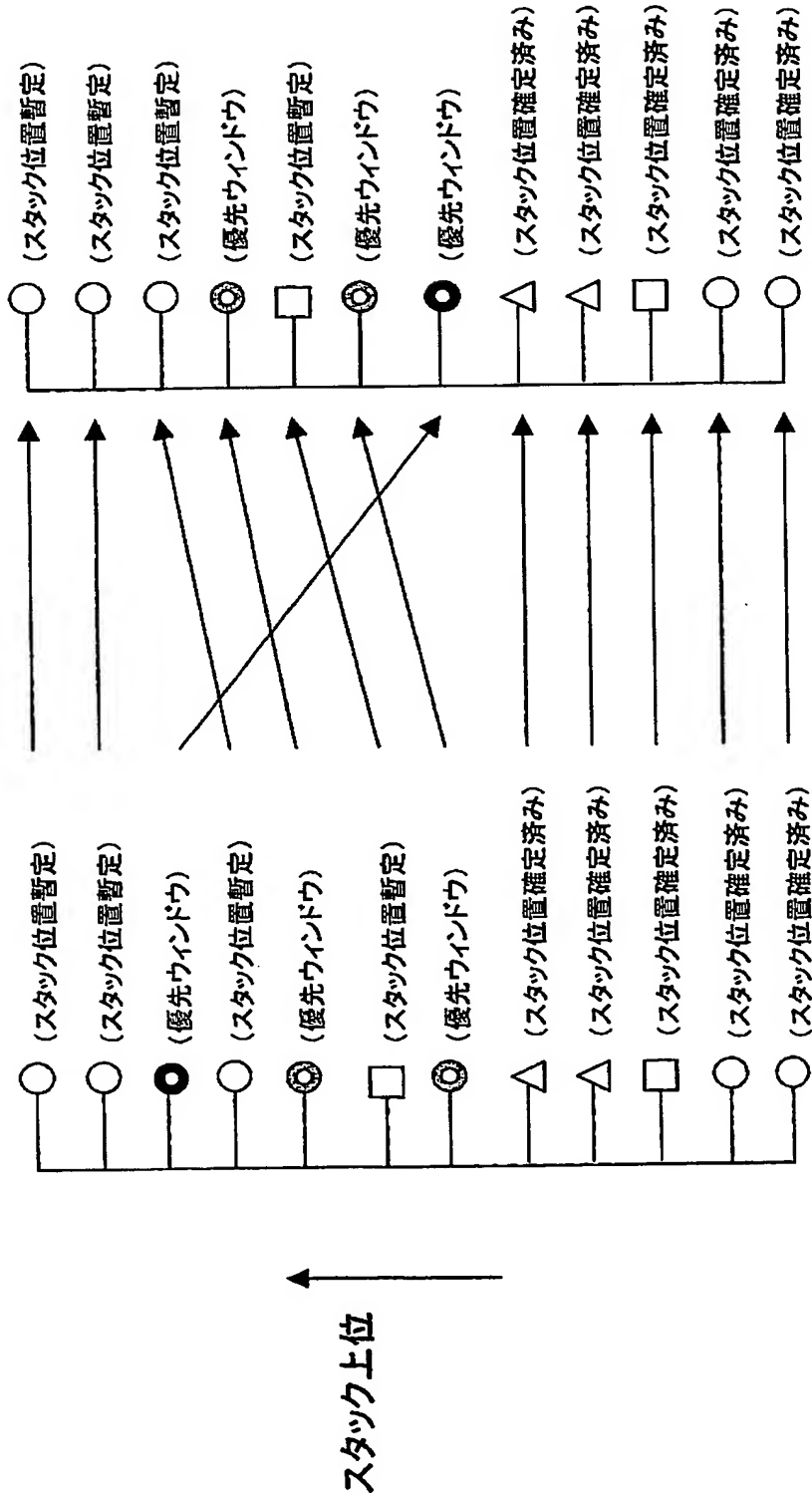
【図 1 1】



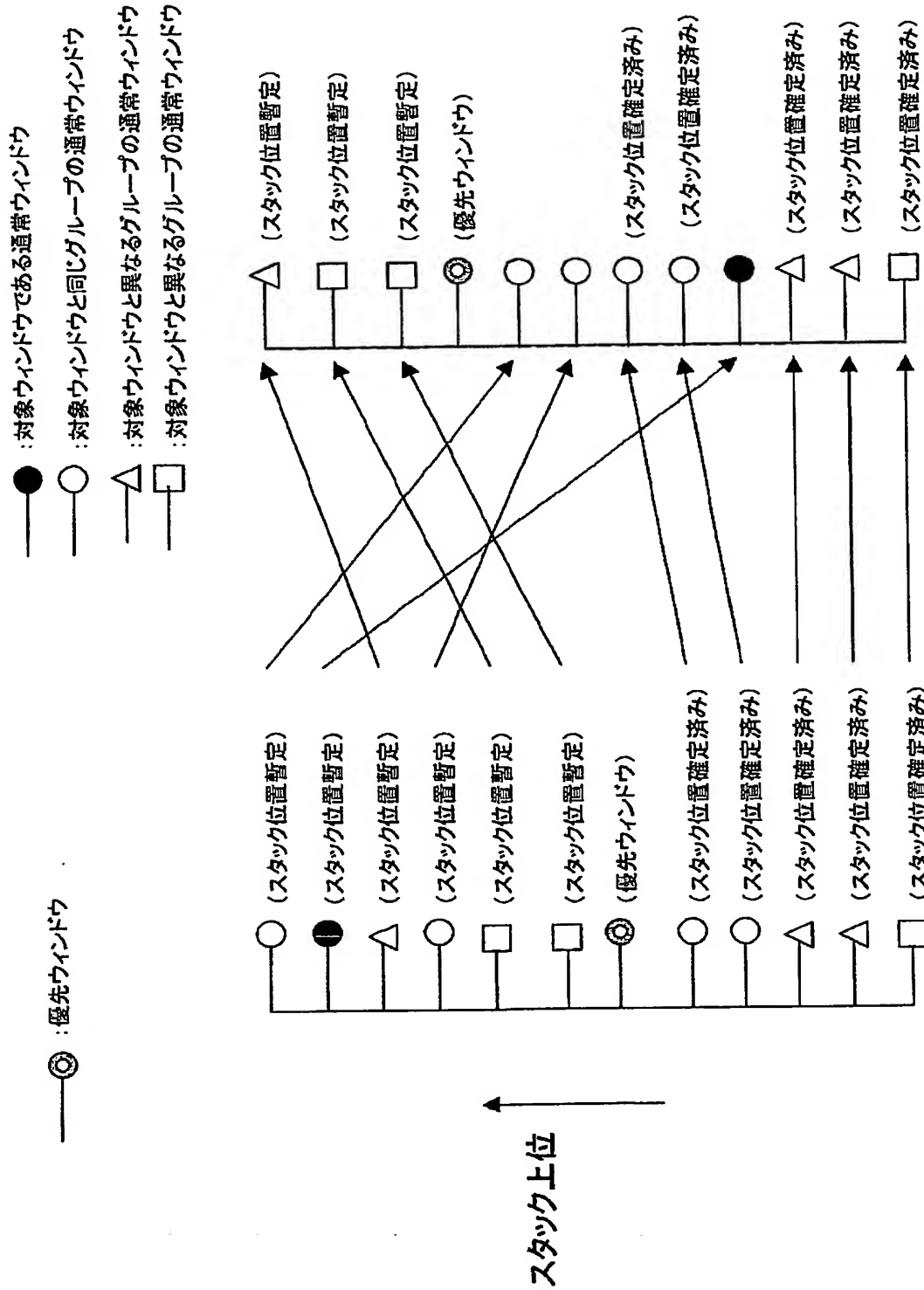
【図 12】

○ : 対象ウィンドウと同じグループの通常ウィンドウ
 △ : 対象ウィンドウと異なるグループの通常ウィンドウ
 □ : 対象ウィンドウと異なるグループの通常ウィンドウ

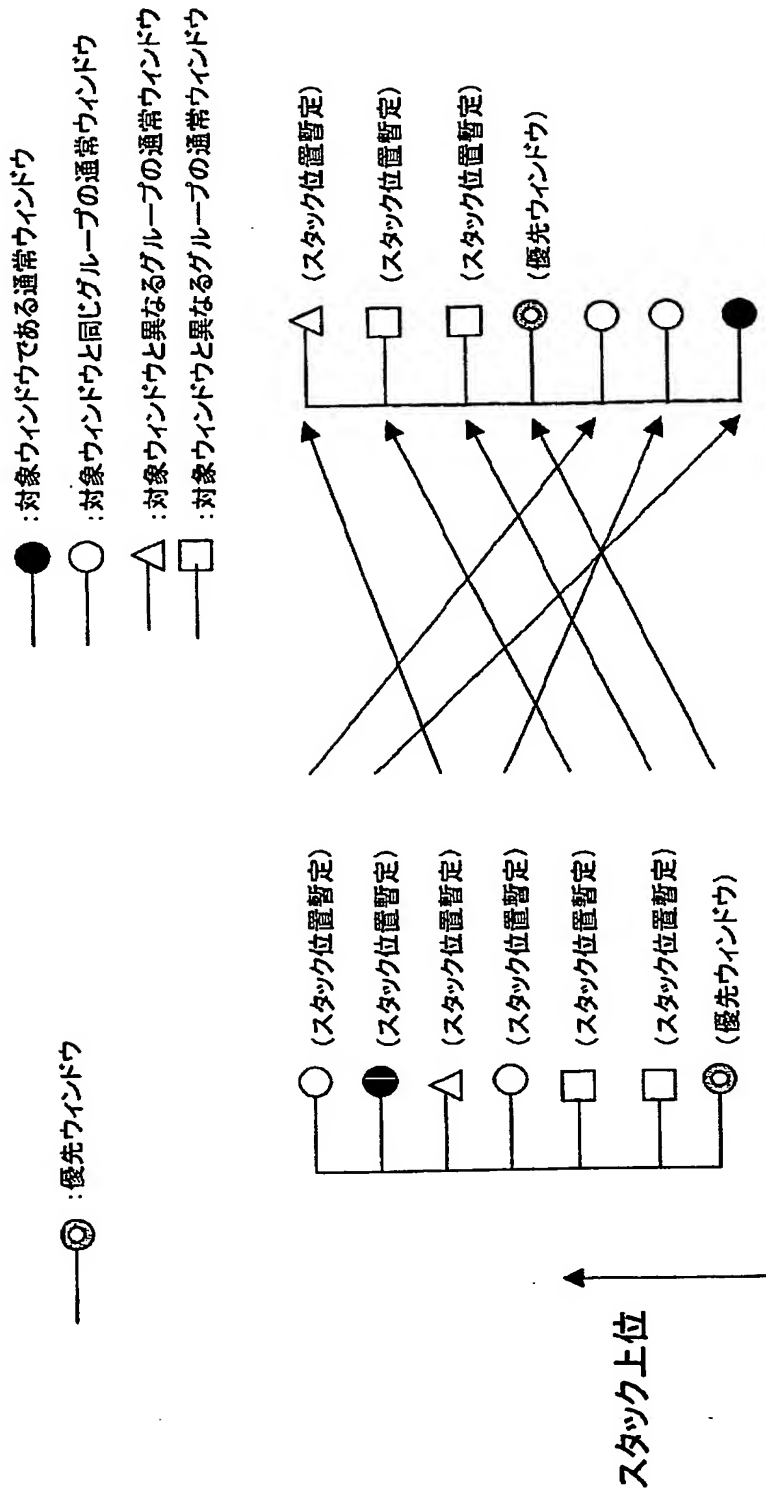
⊙ : 優先ウィンドウ
 ● : 対象ウィンドウである優先ウィンドウ



【図13】



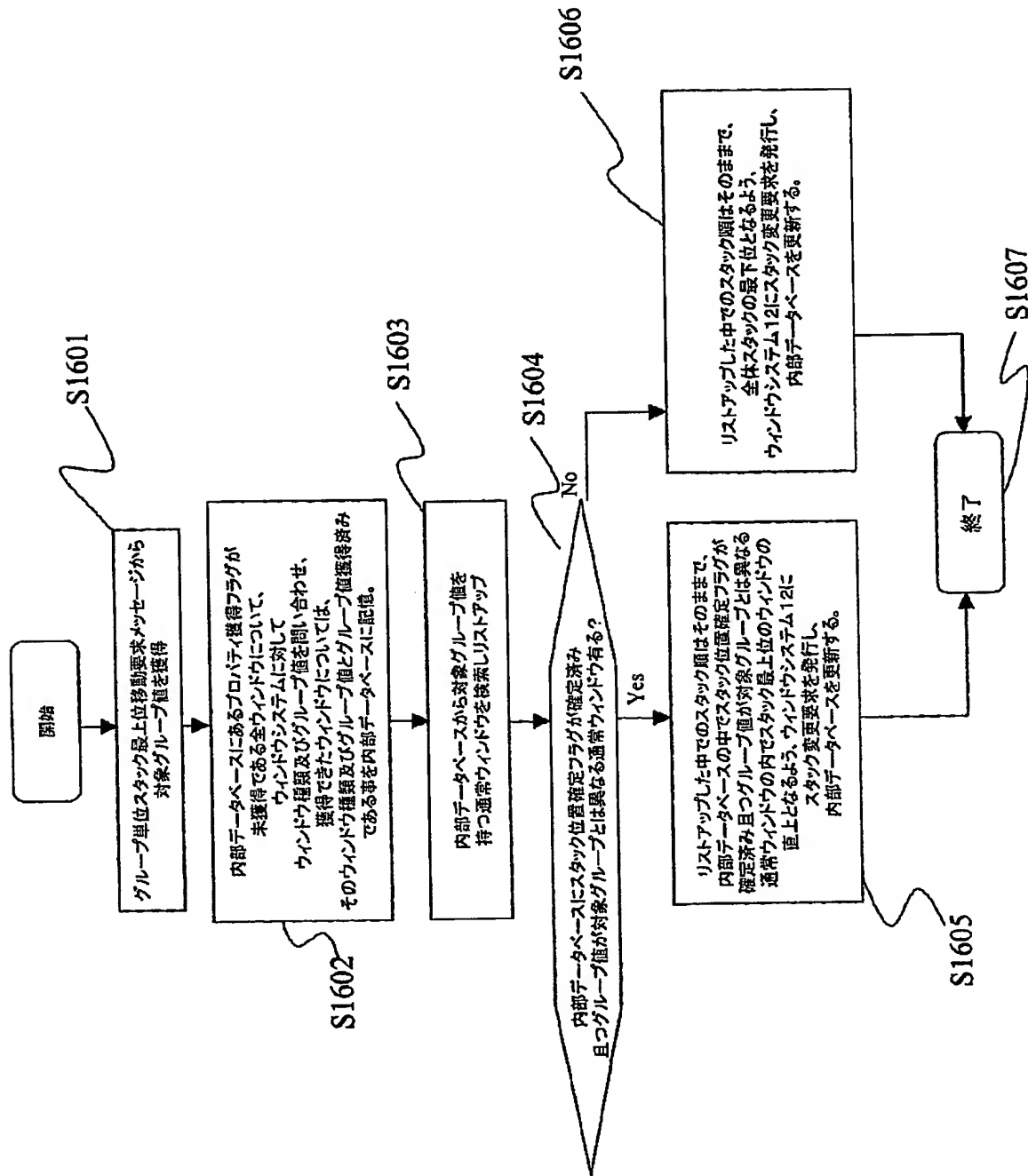
【図 15】



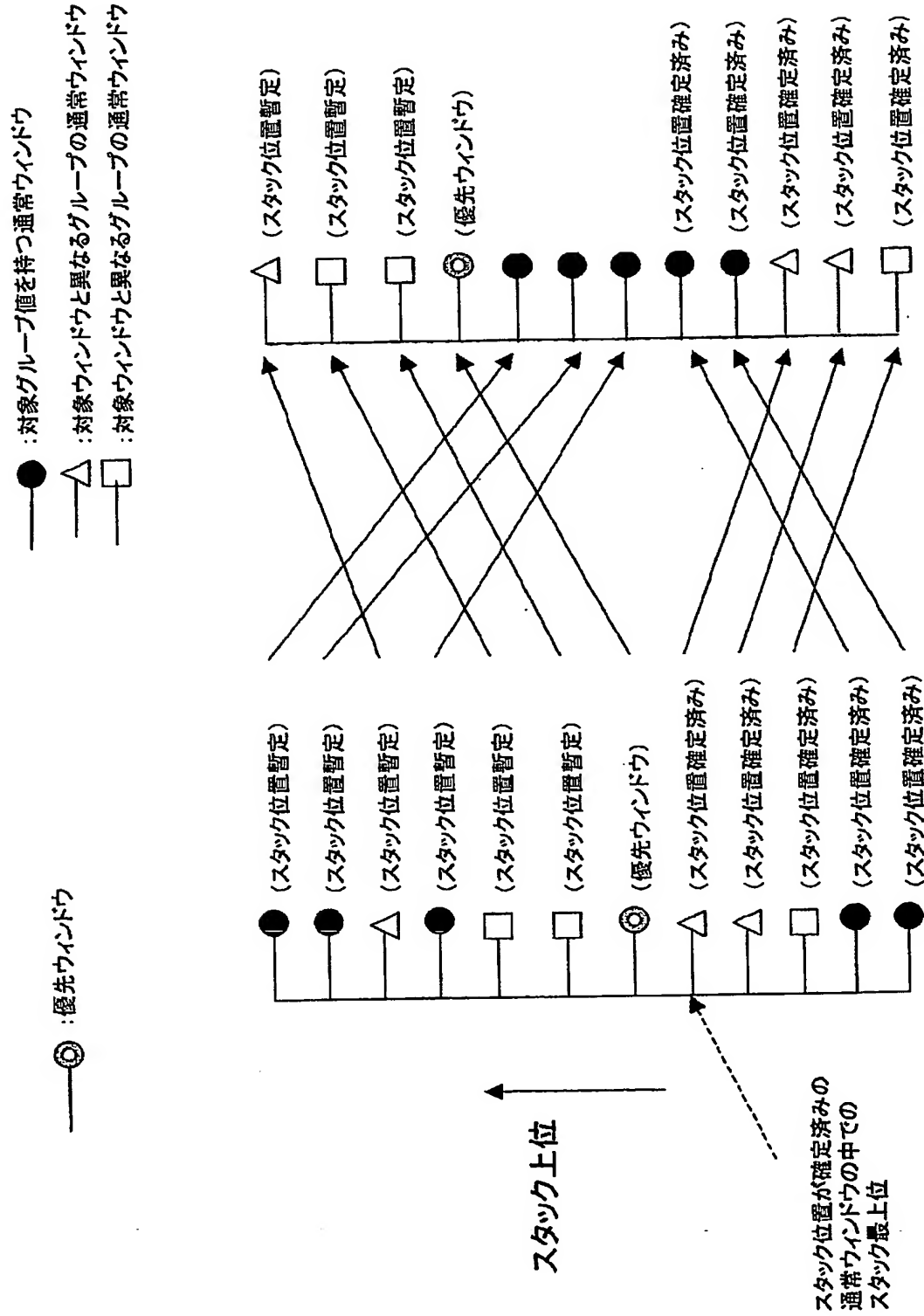
スタック変更後

スタック変更前

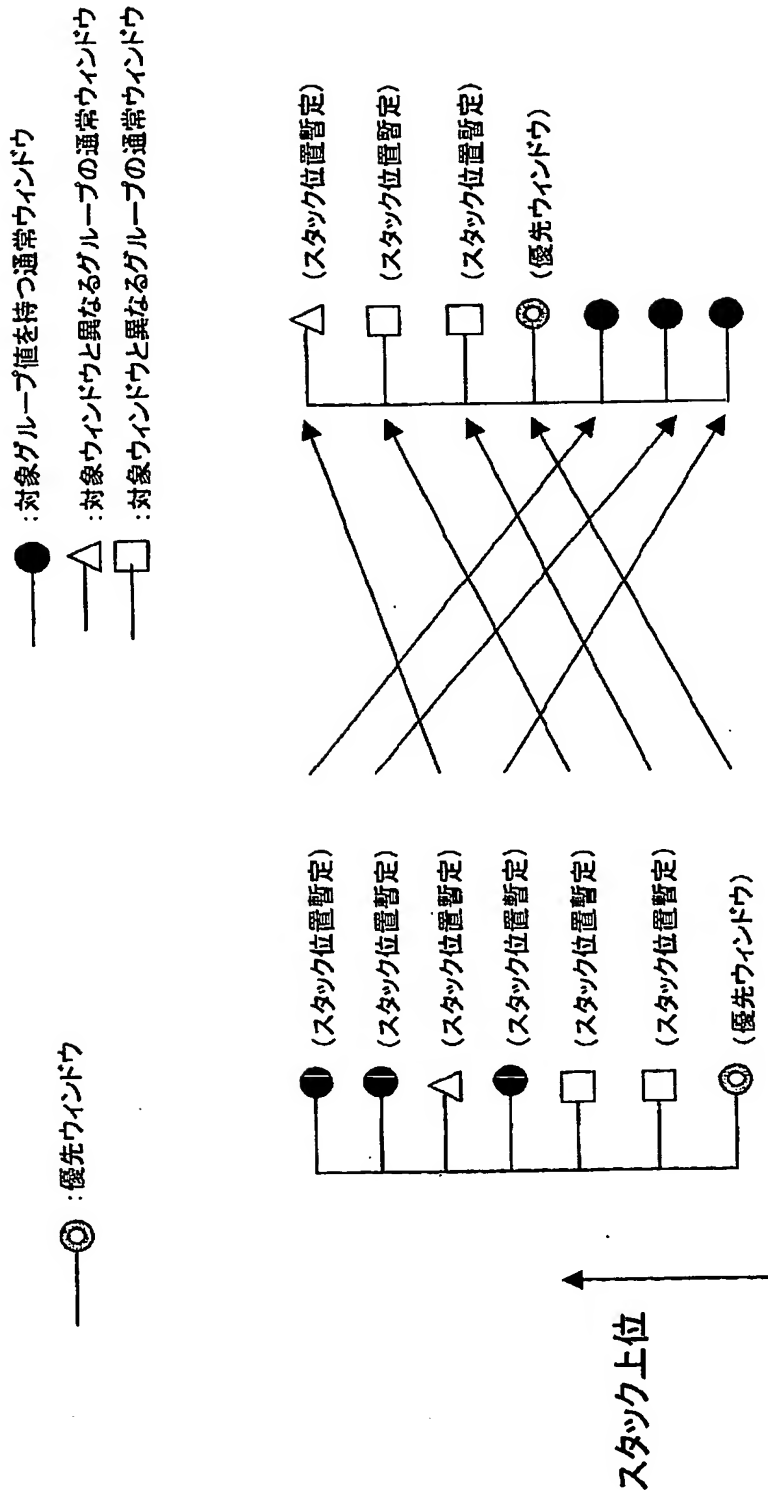
【図16】



【図17】



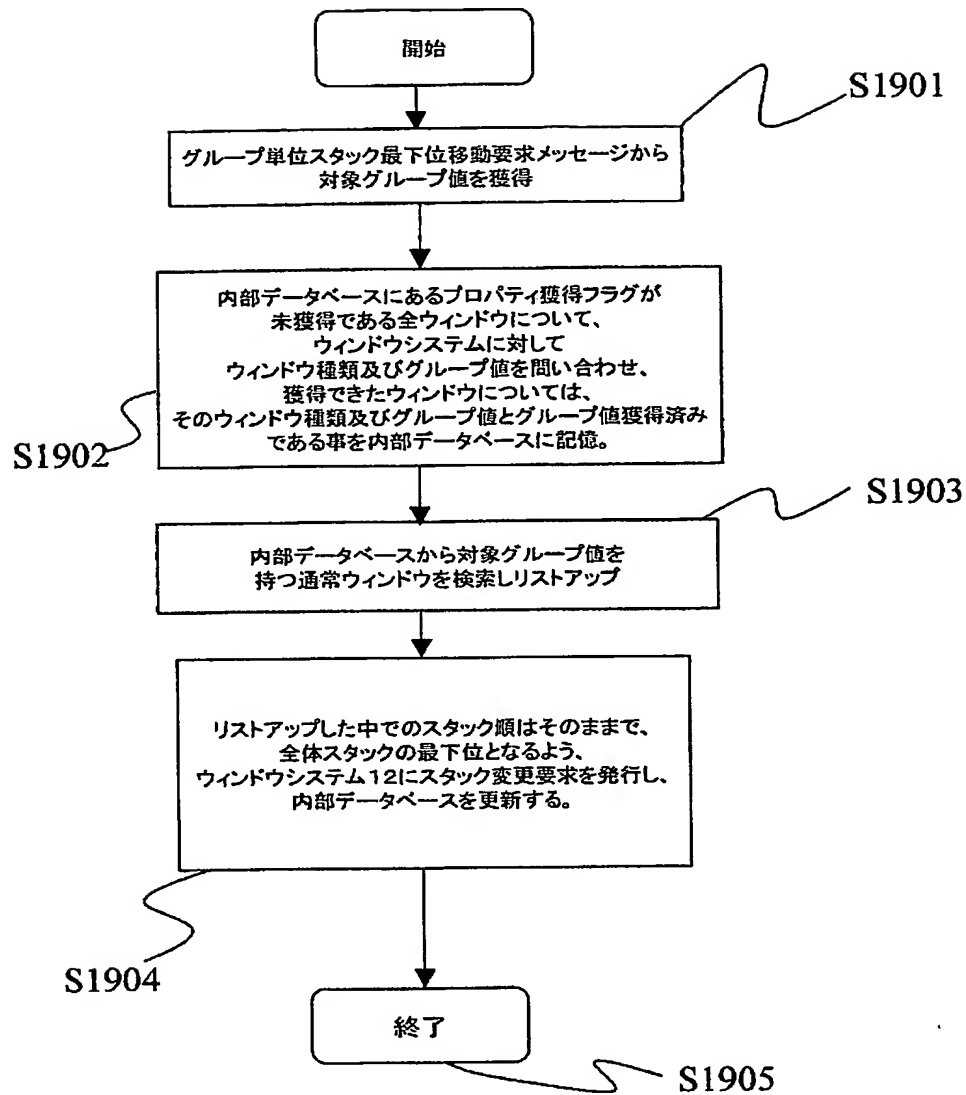
【図 18】



スタック変更後

スタック変更前

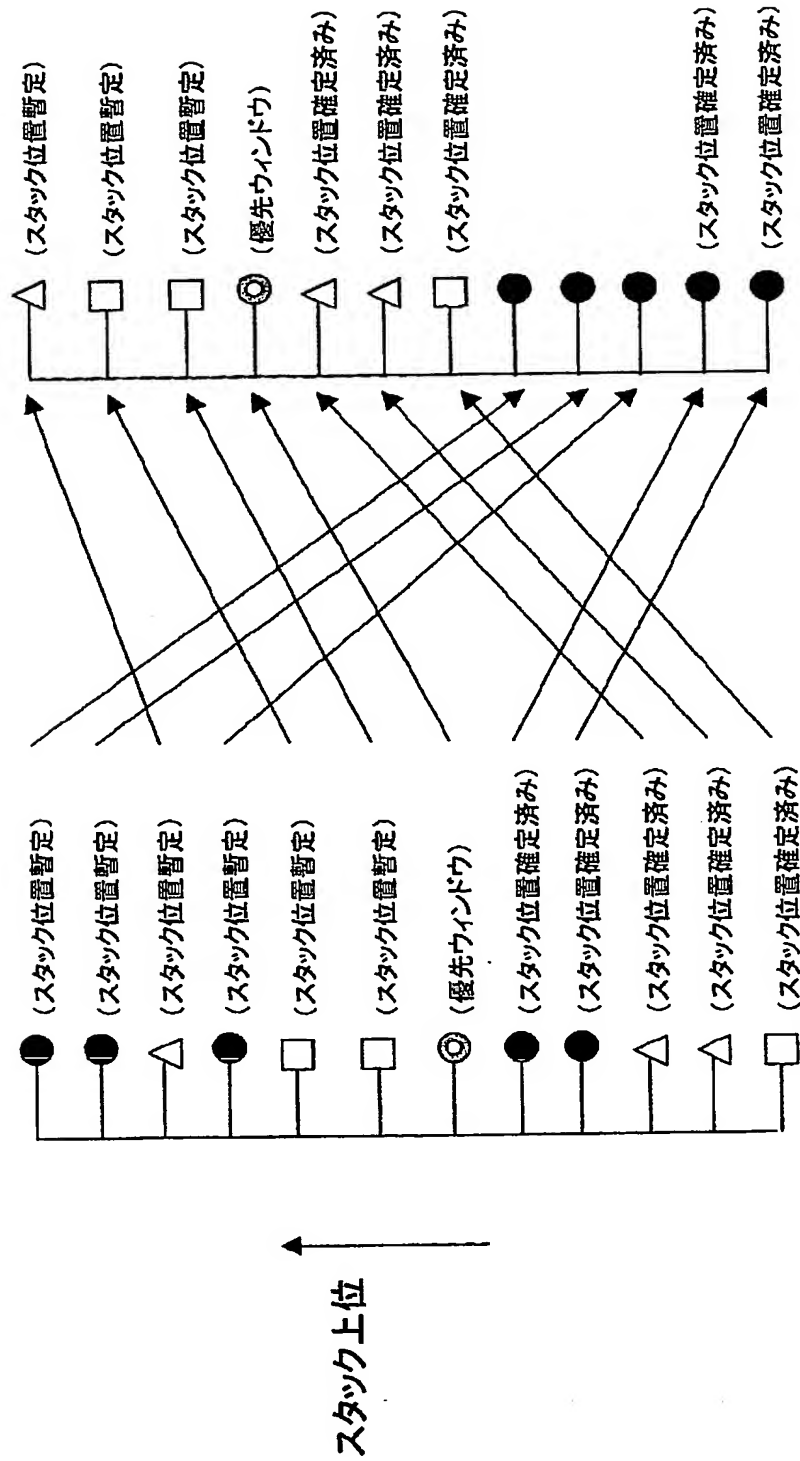
【図 19】



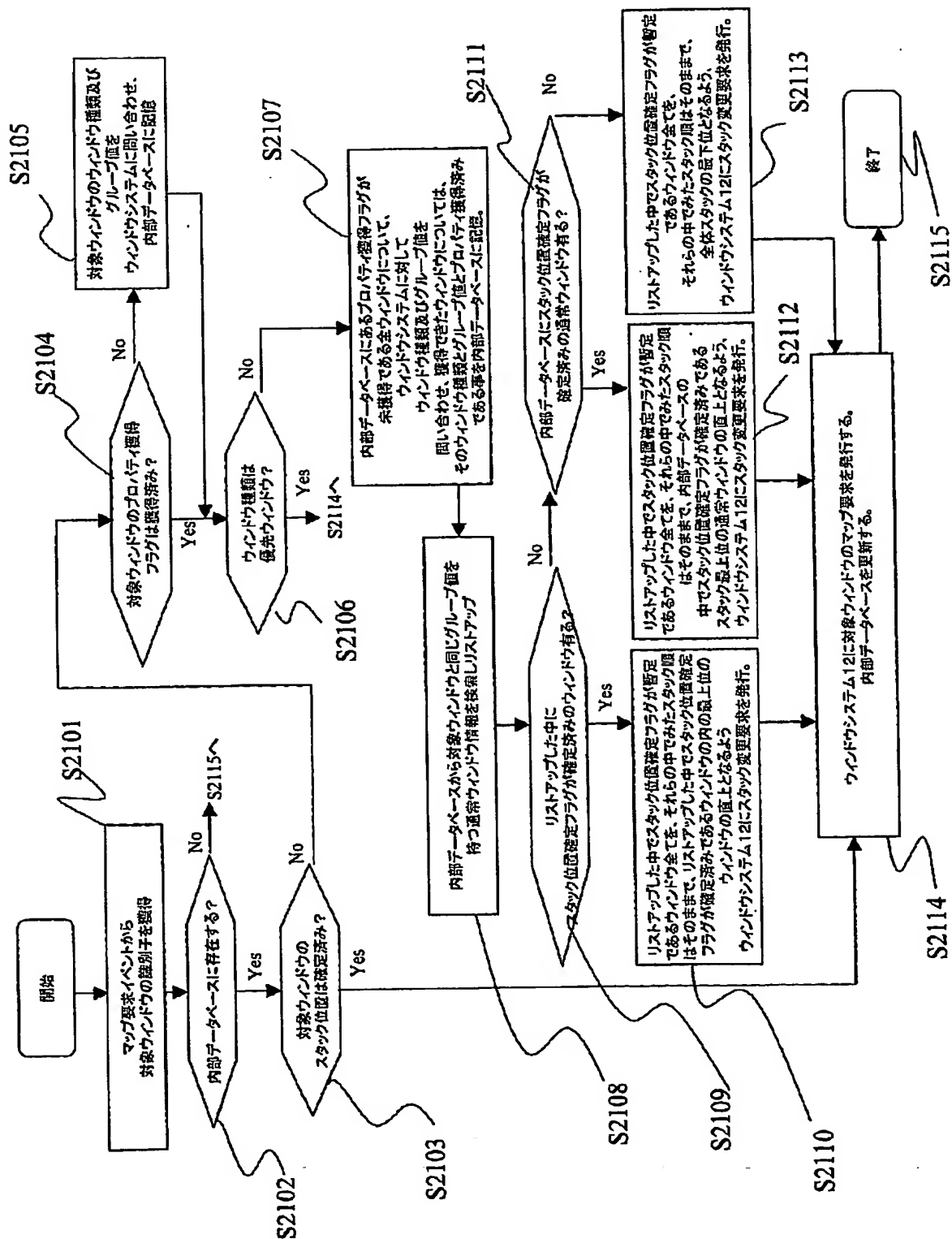
【図 20】

- : 対象グループ値を持つ通常ウィンドウ
- △ : 対象ウィンドウと異なるグループの通常ウィンドウ
- : 対象ウィンドウと異なるグループの通常ウィンドウ

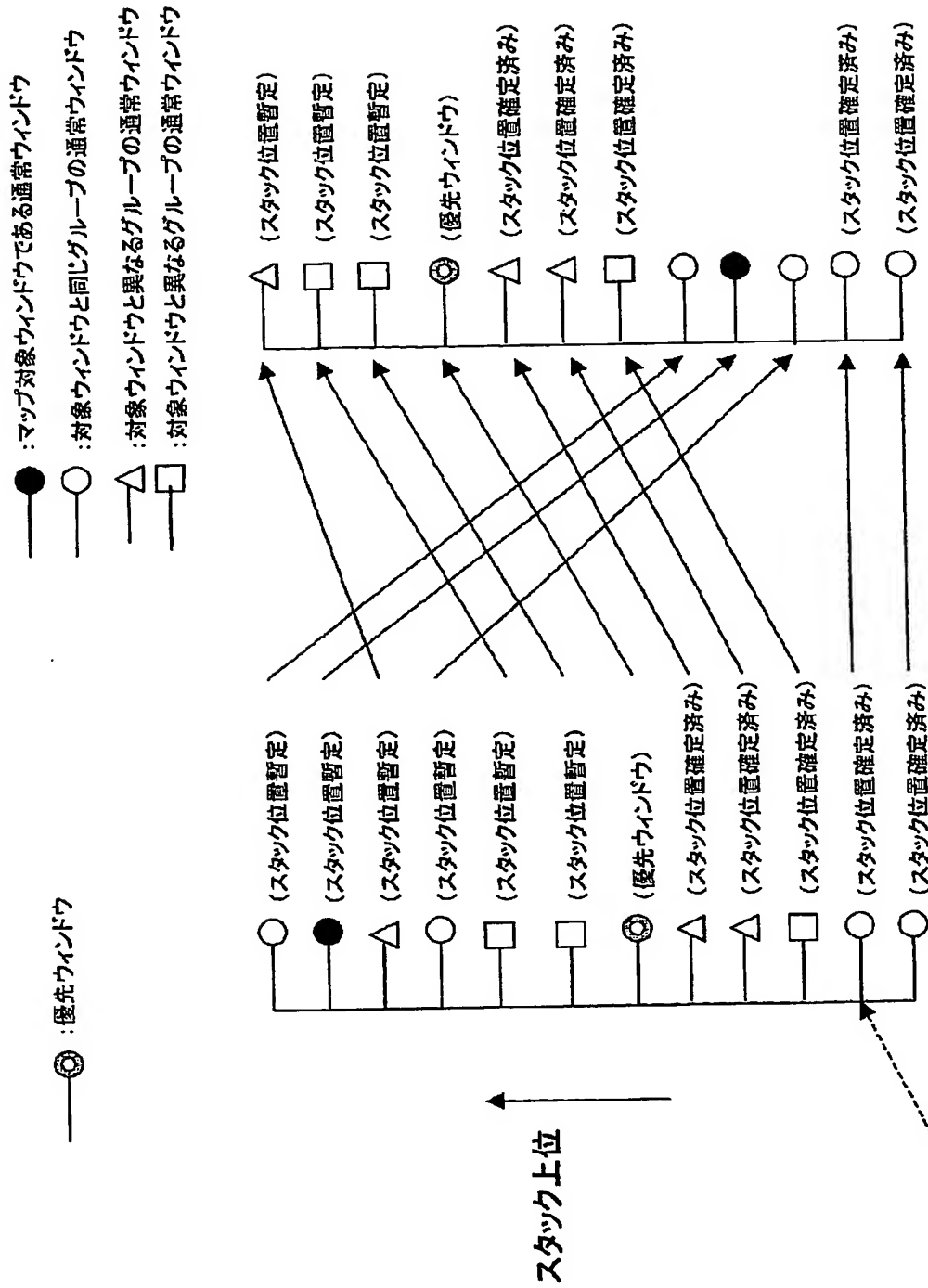
⊙ : 優先ウィンドウ



【図 21】

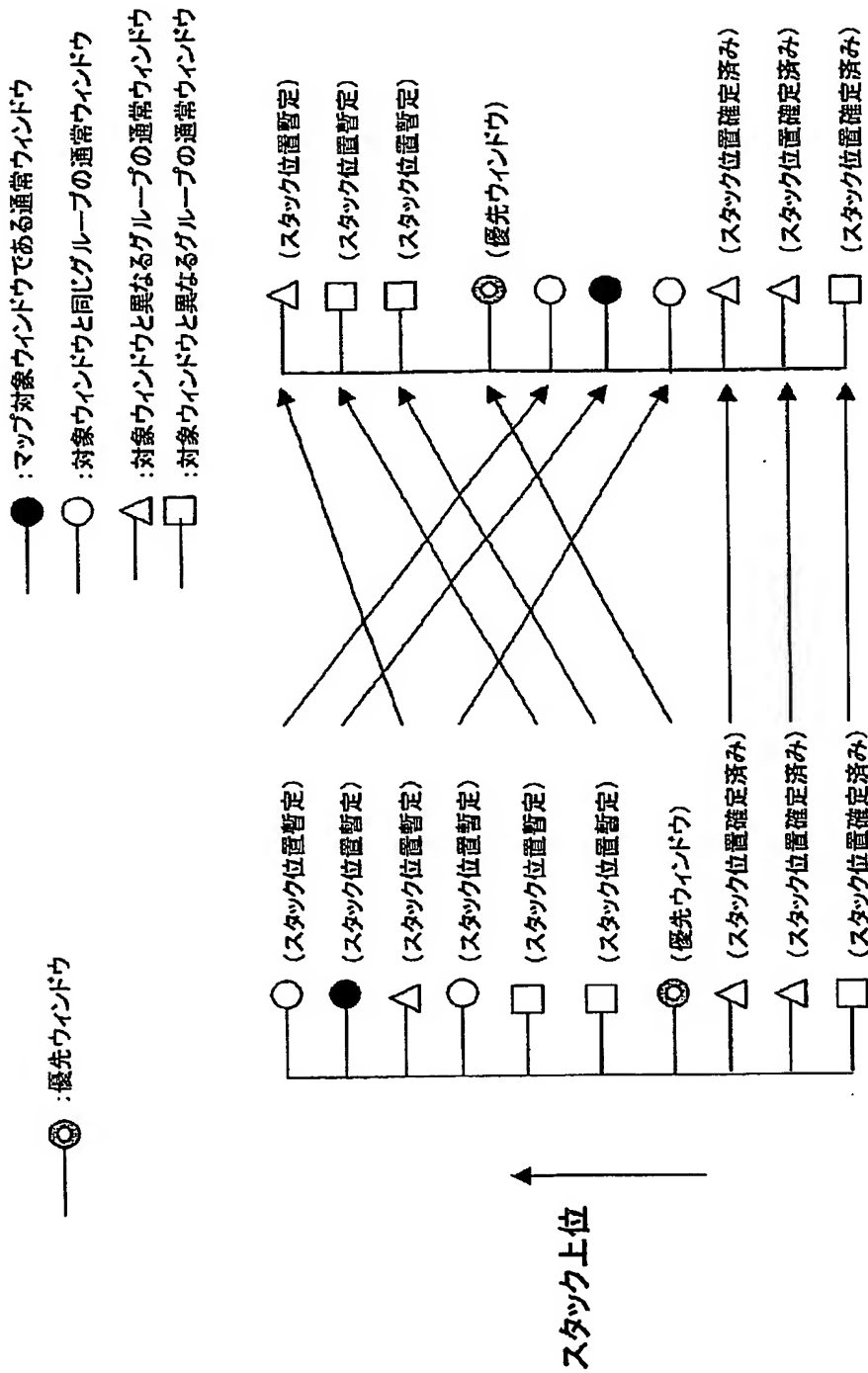


【図 22】



同一グループ且つスタック位置が確定済みの
通常ウィンドウの中のスタック最上位

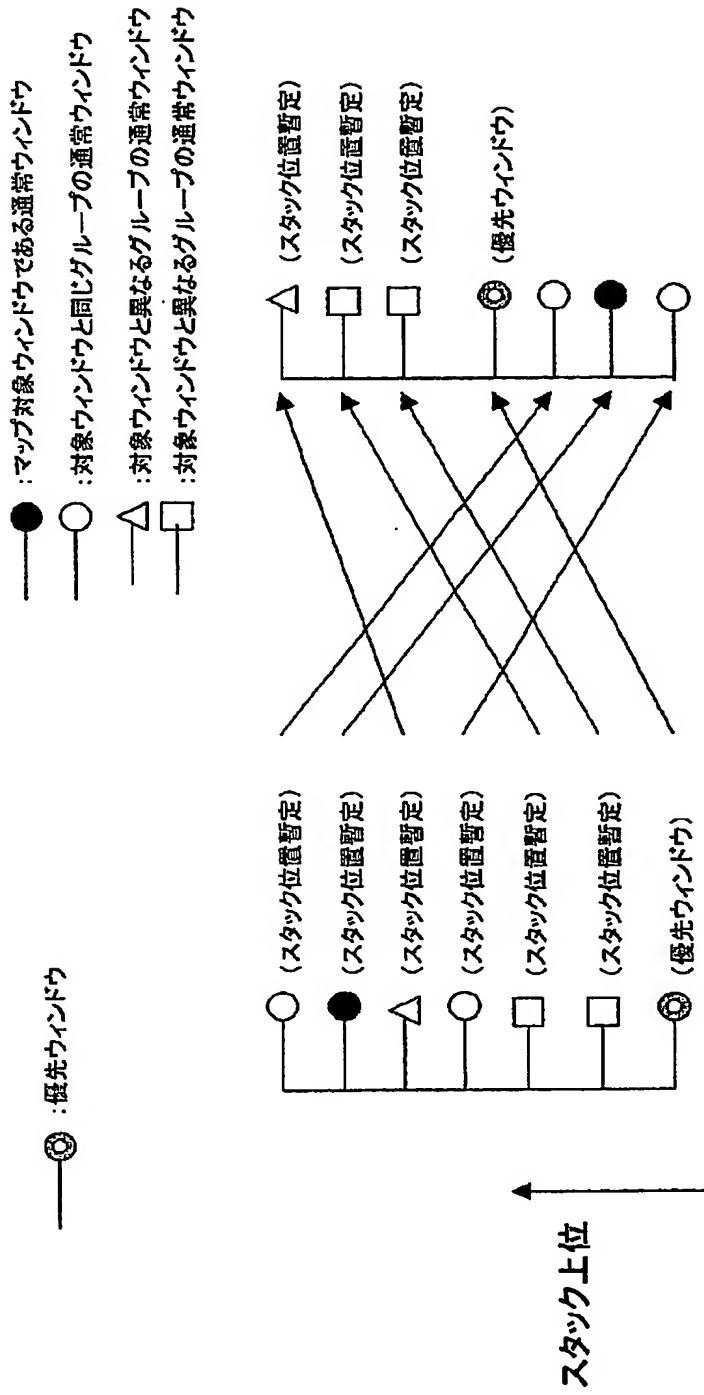
【図 23】



スタック変更後

スタック変更前

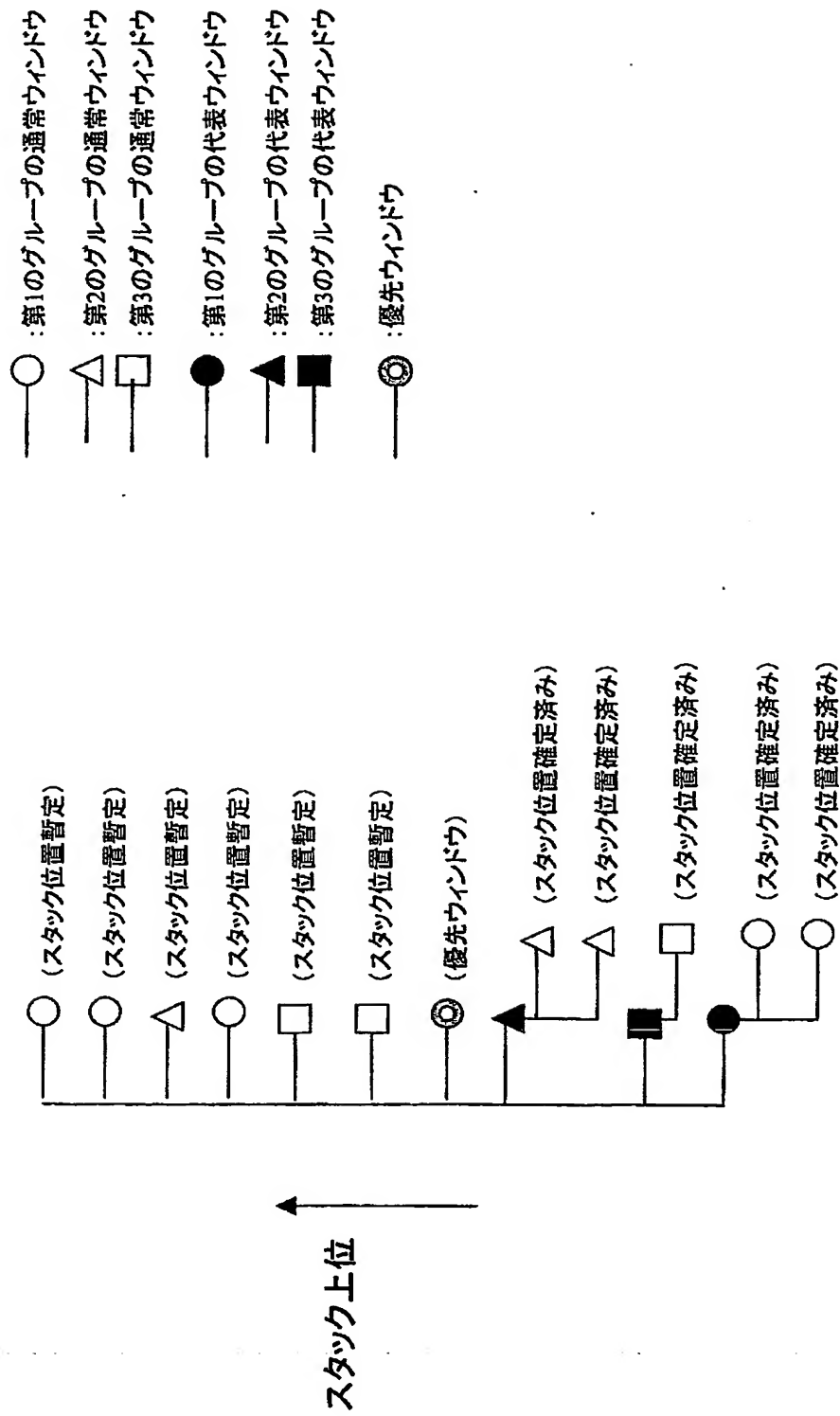
【図 24】



スタック変更後

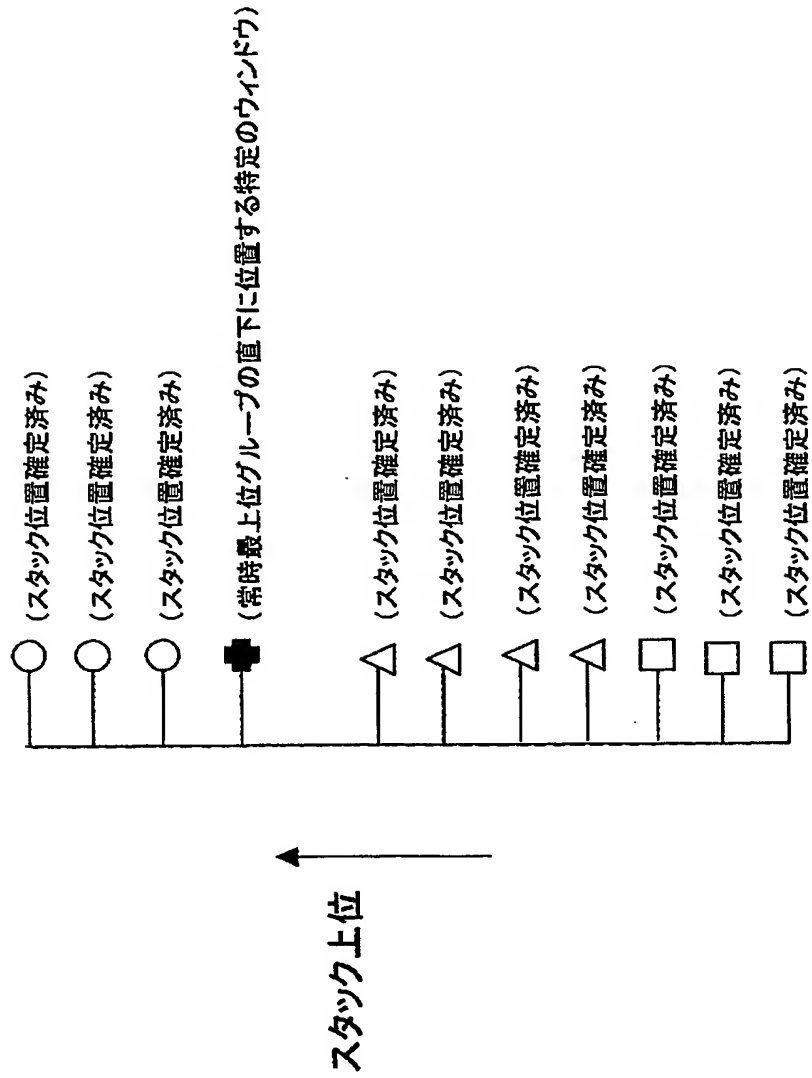
スタック変更前

【図 25】



【図 26】

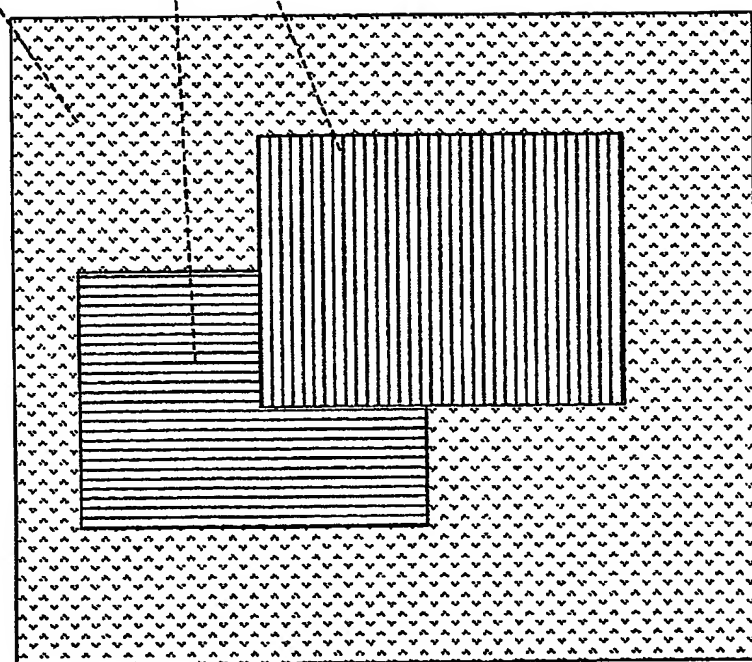
- : 第1のグループの通常ウィンドウ
- △ : 第2のグループの通常ウィンドウ
- : 第3のグループの通常ウィンドウ



【図 27】

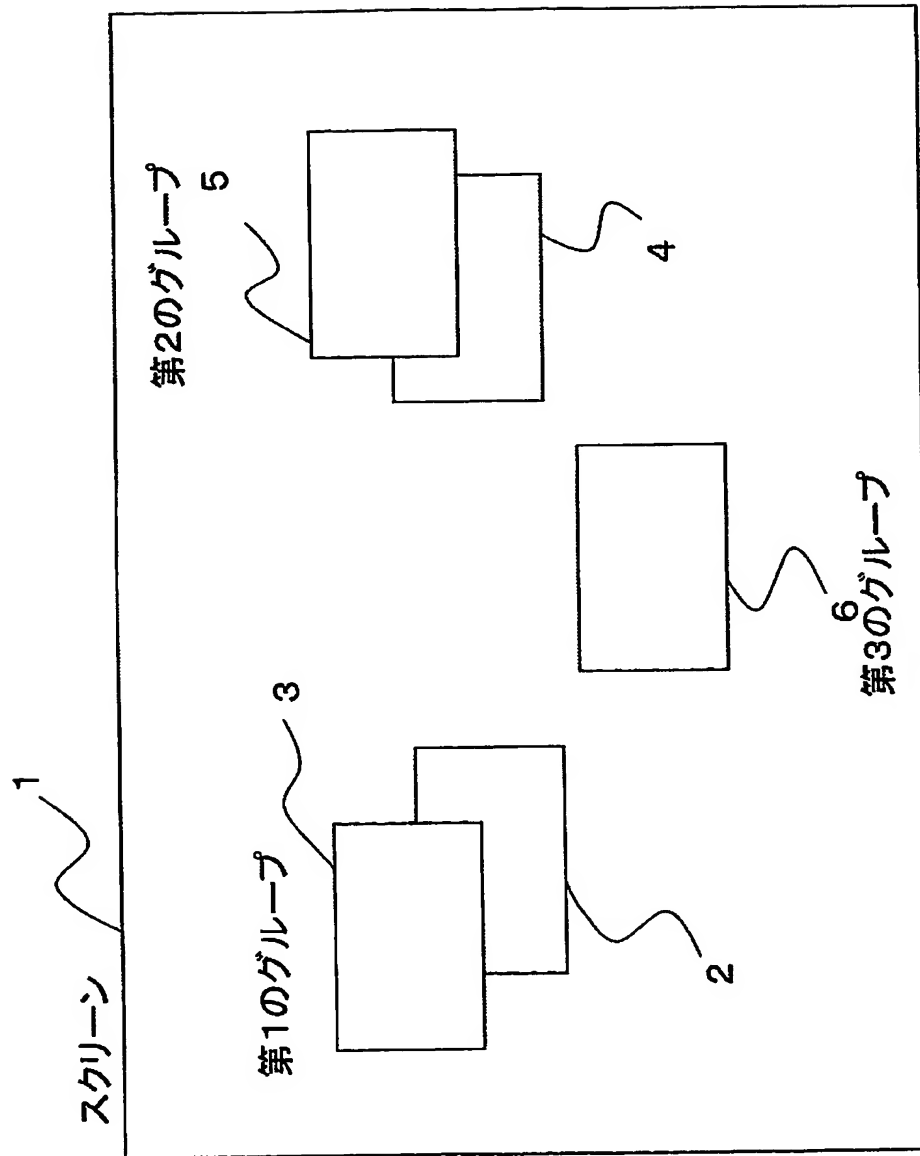
常時、最上位グループの直下に位置する特定のウィンドウ

最上位グループのウィンドウ

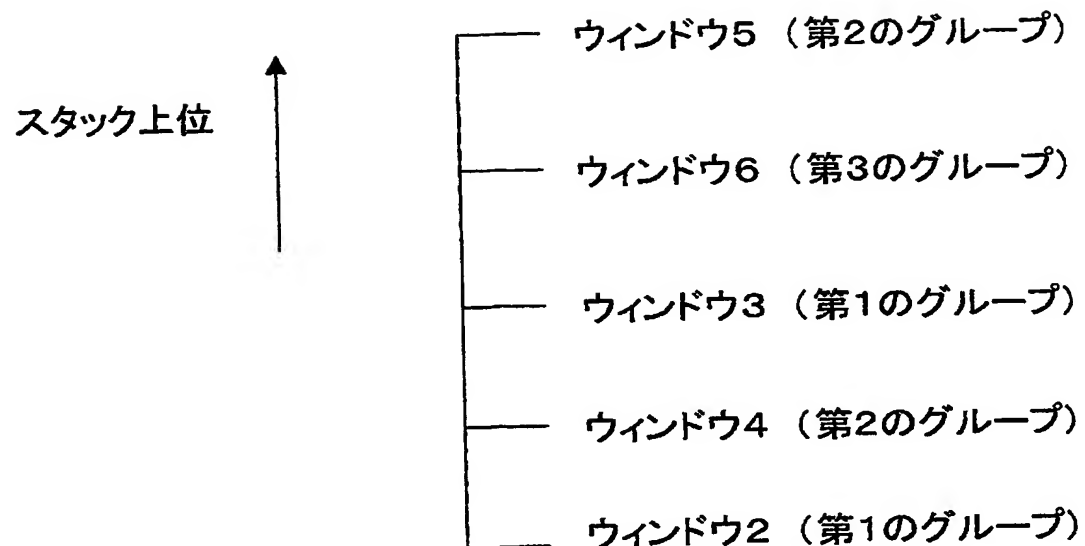


スクリーン

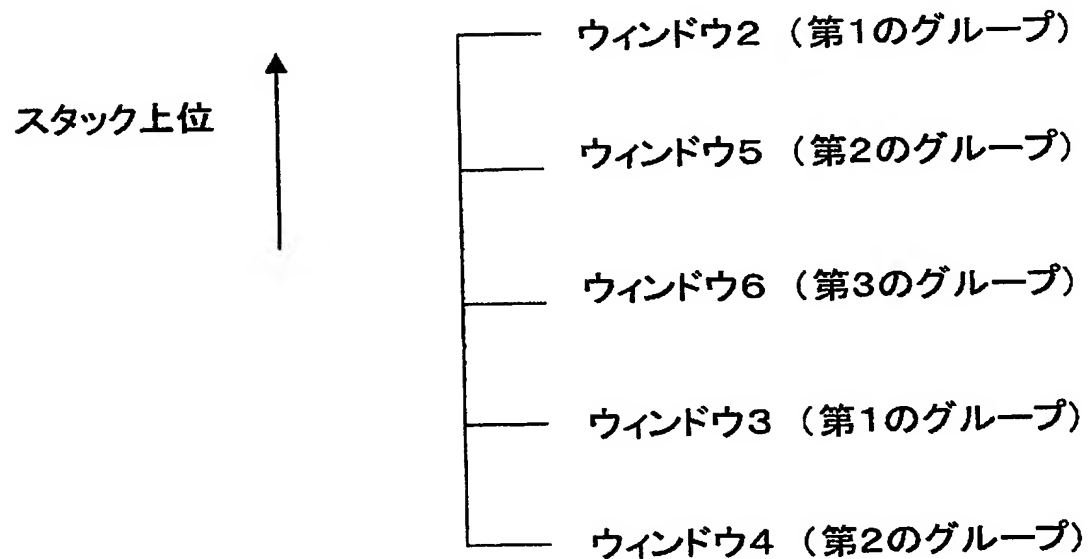
【図 28】



【図 29】



【図 30】



【書類名】 要約書

【要約】

【課題】 ウィンドウスタックの制御において、最前面でウィンドウを表示中のアプリケーションのウィンドウ表示に、他のアプリケーションが不用意に影響を及ぼす事の無いウィンドウスタック制御方法を提供することを目的とする。

【解決手段】 コンピュータから表示装置に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウスタックを制御するに際し、アプリケーションプログラムがウィンドウにグループを指定し、ウィンドウ管理プログラムがウィンドウのスタック順をグループ毎に一連とする。

【選択図】 図 1

特願 2 0 0 3 - 1 0 6 3 9 3

出 願 人 履 歴 情 報

識別番号

[0 0 0 0 0 5 8 2 1]

1. 変更年月日

1 9 9 0 年 8 月 2 8 日

[変更理由]

新規登録

住 所

大阪府門真市大字門真 1 0 0 6 番地

氏 名

松下電器産業株式会社